

# Recommending Tags for Images

Deep Learning Approaches for Personalized Tag Recommendation

A thesis submitted for the degree of  
*Doctor of Natural Science (Dr. rer. nat.)*

In the subject of Computer Science

by

**Thi Hong Hanh Nguyen**

Department of Computer Science  
Information Systems and Machine Learning Lab (ISMLL)

UNIVERSITY OF HILDESHEIM, GERMANY



Summer 2018

---

## Acknowledgements

I am using this opportunity to express my gratitude to everyone who supported me throughout my study at University of Hildesheim. First and foremost, I want to thank my supervisor Prof. Dr. Dr. Lars Schmidt-Thieme for his continuous support, patience, insightful advice and immense knowledge. His guidance helped me in all the time of research and writing of the papers and this thesis. Moreover, I would like to thank the rest of my thesis committee: Prof. Dr.-Ing. Bodo Rosenhahn and Prof. Dr. Klaus Schmid, for their comments encouragement, and hard questions.

I would like to thank all members of the Information Systems and Machine Learning Lab (ISMLL) at University of Hildesheim. We have had a good time period working together, not only in research but also in daily-life activities. I especially thank Martin Wistuba, for his encouragement, sharing ideas, fruitful comments and advice. My thanks also go to Dr. Lucas Rêgo Drumond, Dr. Josif Grabocka, and Nicolas Schilling for stimulating discussion and enlightening me the first glance of research.

I would like to thank the ministry of education and training Vietnam for the financial support during my Ph.D. studies. Many thanks also go to the ISMLL for supporting me a fellowship as well as funding for all my trips to conferences. Moreover, I would like to thank the DAAD (Deutscher Akademischer Austausch Dienst) for supporting me one-year fellowship.

Last but not the least, I would like to express my deep gratitude to my family who gives me strength and encouragement.

## Abstract

Social media has become an integral part of numerous individuals as well as organizations, with many services being used frequently by a majority of people. Along with its widespread use, the amount of information explodes when people use these services. This demands for efficient tools as well as methods to assist data management and retrieval. Annotating resources by keywords, known as the tagging task, is a solution to improve categorizability and findability of resources. However, tagging is a human, time-consuming task, which requires the user's focus to figure out many keywords in a short moment and manually enter them into the system. To encourage users to tag their resources more correctly and frequently, tag recommendation is adopted into the social tagging systems to suggest relevant keywords for resources.

In this thesis, we will address the problem of personalized tag recommendation for images and present ways to solve this problem by combining the advantages of the user relation with the images' content. In order to suggest tags for unobserved images, their visual contents are used to replace the index-based information of the image entity in the tagging relations. Because the limitation of low-level features does not show the "content" of images, we propose to utilize a deep learning based approach to learn high-level visual features concurrently with the scoring-tag estimator. For the tag predictor, a latent factor model or a multi-layer perceptron is selected to compute scores of tags by which the top selected tags are sorted in descending order. As a further development upon our findings, we examine the inside and outside context of images to enhance the accuracy of estimators. Regarding the image-inside context, we are motivated by the fact that objects, such as cars or cats are influential on the user's selection criteria. Regarding the image-outside context, the image's surrounding text contributes to the clarity of the image's content for different users. We consider these contextual features as a supporting part which is combined with the mainly visual representation to enhance the tag recommendation performance. Finally, as an additional technique, transfer learning is also adapted to support the proposed models to overcome the limitations of too small training data and boost up their performance. This thesis demonstrates the usefulness and versatility of deep learning approaches for tag recommendation and highlights the importance of the learned image's content in predicting personalized tags. Directions for future work include semantic enhancements to context-based representation and extensions of the content-aware approaches to different recommendation scenarios.



## Abstrakt

Die sozialen Medien sind zu einem integralen Bestandteil zahlreicher Einzelpersonen und Organisationen geworden, gemessen an der Häufigkeit mit der die Dienste von einer Mehrheit von Menschen genutzt werden. Zusammen mit seiner weitverbreiteten Verwendung explodiert die Menge an Informationen nahezu, wenn viele Menschen diese Dienste nutzen. Dies erfordert effiziente Tools sowie Methoden zur Unterstützung des Datenmanagements und des Wiederfindendauffindens von Inhalten. Das Annotieren von Ressourcen durch Schlüsselwörter, auch Tagging genannt, bietet eine Lösung die Kategorisierbarkeit und Auffindbarkeit von Ressourcen zu verbessern. Es ist jedoch eine Aufgabe die von Menschen ausgeführt wird und von diesem verlangt, passende Tags innerhalb von einer kurzen Zeit zu finden. Um sowohl die Qualität aber auch die Quantität an Tags zu verbessern werden Empfehlungssysteme verwendet, die Tags automatisch vorschlagen, die zu dem gegebenen Kontext passen.

In dieser Dissertation adressieren wir das Problem der personalisierten Tag-Empfehlung für Bilddaten und zeigen Lösungswege auf, die darauf basieren die Interaktion des Benutzers mit dem Bildinhalt zu modellieren. Um Tags für bisher nicht observierte Bilder zu empfehlen nutzen wir die Information des Bildes direkt innerhalb unseres Modells. Dafür schlagen wir vor Faktorisierungsmodelle oder neuronale Netze als Scoring Funktion für die entsprechenden Bilder zu lernen. Da Low-Level Features der Bilder nicht präzise den Inhalt des Bildes abbilden, verwenden wir Methoden des Deep Learnings um direkt für das Problem bessere Abstraktionen zu lernen. Darüber hinaus verwenden wir den Kontext innerhalb als auch ausserhalb des Bildes, indem wir zum Einen Methoden der Objekterkennung, beispielsweise indem Objekte wie Autos oder Katzen abgebildet sind, als auch den äußeren Kontext der Bilder in Form von Kommentaren oder Überschriften erkennen und nutzen.

Beide Aspekte des Kontexts verbinden wir als Unterstützung zur eigentlichen Bildklassifikation um die Güte der Empfehlungen zu verbessern. Abschließend zeigen wir, das Methoden des Transferlernens sowohl Probleme wie zu geringe Datenmengen in der Zieldomäne löst und die Performanz im Allgemeinen verbessert. Diese Arbeit zeigt die Nützlichkeit von Deep Learning Methoden als Empfehlungssysteme, und hebt dabei die Bedeutung des Bildinhalts hervor. Zukünftige Arbeit in diesem Gebiet umfasst neben besseren Feature Repräsentationen von Text und Bildern auch die Anwendung bestehender Modelle in anderen Szenarien von Empfehlungssystemen.

## Notation

This thesis will adhere to the following notation and nomenclature:

$\mathbb{N}$	the set of natural numbers
$\mathbb{R}$	the set of real numbers
$\mathbb{R}^m$	the set of $m$ -dimensional vectors over $\mathbb{R}$
$\mathbb{R}^{n \times m}$	the matrix of size $n \times m$ over $\mathbb{R}$
$\mathbb{R}^{n \times m \times c}$	the 3-D tensor of size $n \times m \times c$ over $\mathbb{R}$
$\{0, 1\}^m$	the set of $m$ -dimensional binary vectors
$\mathbf{X}$	matrix
$(\mathbf{X})'$	transposed matrix
$\ \mathbf{X}\ $	matrix norm (subscript if any denotes what norm)
$X_{i,j}$	the element at the position $(i, j)$ of the matrix $\mathbf{X}$
$\mathbf{X} \odot \mathbf{Y}$	Hadamard (elementwise) product between matrices $\mathbf{X}$ and $\mathbf{Y}$
$\mathbf{x}$	vector
$(\mathbf{w})'$	transposed vector
$\mathbf{x}_i$	vector indexed for some purpose
$(x_i)_j$	the $j$ -th element of the vector $\mathbf{x}_i$
$w_j$	the $j$ -th element of the vector $\mathbf{w}$
$a$	scalar
$\langle \mathbf{v}_i, \mathbf{v}_j \rangle$	$= \sum_{k=1}^d (v_i)_k (v_j)_k$ (inner product)
$\mathcal{P}$	dataset
$\mathcal{P}^{train}, \mathcal{P}^{test}$	training, and test sets respectively
$ \mathcal{P} $	the number of instances in the set $\mathcal{D}$
$\mathcal{L}$	loss function for some purpose
$f$	function for some purpose
$\frac{\partial f(\mathbf{w})}{\partial w_i}$	the partial derivative of a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ at $\mathbf{w}$ w.r.t. $w_i$
$\operatorname{argmax}_{x \in C} f(x)$	the set $\{x \in C : f(x) \geq f(y), \forall y \in C\}$
$\operatorname{argmax}_{t \in T, K} (\hat{y}_{u,i})_t$	the set of $K$ tags $t \in T$ having the largest values

## List of commonly used abbreviations

AIA	Automatic Image Annotation
AUC	Area Under the ROC Curve
BPR	Bayesian Personalized Ranking
CNN	Convolutional Neural Network
DL	Deep Learning
FM	Factorization Machine
IC	Image Classification
PITF	Pairwise Interaction Tensor Factorization
PTR	Photo Tag Recommendation
MF	Matrix Factorization
MLP	Multi-Layer Perceptron
MP	Most Popular Tags
MP-u	Most popular tags by users
NN	Neural Network
OD	Object Detection
RGB	Red, Green, Blue
ROC-curve	Receiver Operating Characteristic curve
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
WARP	Weighted Approximate Ranking
WE	Word Embedding
tf-idf	term frequency-inverse document frequency
w.r.t.	with respect to

---

# Contents

<b>1</b>	<b>Introductory Material</b>	<b>1</b>
1.1	Social Media . . . . .	1
1.2	The Demand for Tag Recommendation . . . . .	2
1.3	Unveiling the Power of Deep Learning . . . . .	4
1.4	Summary of Problems Addressed . . . . .	5
1.5	Main Research Contributions . . . . .	5
1.6	Publications . . . . .	6
1.7	Chapter Overview . . . . .	7
<b>2</b>	<b>Technical Basics</b>	<b>9</b>
2.1	Supervised Learning . . . . .	9
2.2	Deep Feedforward Network . . . . .	10
2.3	Convolution Neural Network . . . . .	12
2.4	Factorization Model . . . . .	15
2.5	Transfer Learning . . . . .	17
<b>3</b>	<b>Problem Description</b>	<b>19</b>
3.1	Personalized Tag Recommendation . . . . .	19
3.2	Personalized Content-aware Tag Recommendation . . . . .	22
3.3	Personalized Content-aware Tag Recommendation with Transferring Knowledge and Image-based Context . . . . .	23
<b>4</b>	<b>Datasets and Pre-Processing</b>	<b>25</b>
4.1	Flickr Dataset . . . . .	25
4.1.1	NUS-WIDE Dataset . . . . .	26
4.1.2	Flickr-PTR Dataset . . . . .	28
4.2	Knowledge-Transfer Dataset . . . . .	29
4.3	Data Preprocessing . . . . .	30
<b>5</b>	<b>Factorization Content-Aware Tag Recommendation</b>	<b>33</b>
5.1	Introduction . . . . .	33
5.2	Review of Tag Recommendation . . . . .	34

## CONTENTS

---

5.3	Problem Extension . . . . .	36
5.4	Content-Aware Factorization Model for Tag Recommendation . . . . .	37
5.4.1	Factorization Machines . . . . .	37
5.4.2	Convolutional neural networks . . . . .	38
5.4.3	Content-aware Factorization Machines . . . . .	39
5.4.4	Optimization . . . . .	41
5.5	Evaluation . . . . .	43
5.5.1	Dataset . . . . .	44
5.5.2	Experimental Setup . . . . .	44
5.5.3	Results . . . . .	45
5.6	Discussion . . . . .	47
<b>6</b>	<b>Personalized Deep Learning for Tag Recommendation</b>	<b>49</b>
6.1	Introduction . . . . .	49
6.2	Problem Extension . . . . .	51
6.3	Personalized Content-aware Tag Recommendation . . . . .	52
6.3.1	Convolutional Neural Network as the Extractor . . . . .	52
6.3.2	Personalized Fully-connected Layer . . . . .	53
6.3.3	Multilayer Perceptron as the Predictor . . . . .	54
6.3.4	Personalized Convolutional Layer . . . . .	54
6.3.5	Optimization . . . . .	55
6.4	Evaluation . . . . .	56
6.4.1	Dataset . . . . .	57
6.4.2	Experimental Setup . . . . .	57
6.4.3	Results . . . . .	58
6.5	Discussion . . . . .	62
<b>7</b>	<b>Personalized Tag Recommendation for Images Using Deep Transfer Learning</b>	<b>63</b>
7.1	Introduction . . . . .	63
7.2	Background Review . . . . .	65
7.3	Problem Extension . . . . .	66
7.4	The Proposed Architecture . . . . .	67
7.4.1	Visual Feature Extraction . . . . .	67
7.4.2	Object Detection . . . . .	69
7.4.3	Factorization Models . . . . .	71
7.4.4	Factorization Models for Image-Aware Tag Recommendation . . . . .	72
7.4.5	Personalized Multilayer Perceptron for Image-Aware Tag Recommendation . . . . .	73
7.4.6	Optimization . . . . .	74
7.5	Evaluation . . . . .	75
7.5.1	Dataset . . . . .	75
7.5.2	Experimental Setup . . . . .	76

7.5.3	Results . . . . .	76
7.6	Discussion . . . . .	79
<b>8</b>	<b>Personalized Tag Recommendation Using Text Transfer Learning</b>	<b>81</b>
8.1	Related Work . . . . .	82
8.2	Problem Extension . . . . .	83
8.3	The Proposed Approach . . . . .	84
8.3.1	Visual Feature Extraction . . . . .	85
8.3.2	Textual Feature Extraction . . . . .	87
8.3.3	Object-Detected Feature Extraction . . . . .	91
8.3.4	Factorization Models . . . . .	92
8.3.5	Factorization Machine with Tag Features . . . . .	93
8.3.6	Personalized Neural Networks . . . . .	93
8.3.7	Optimization . . . . .	98
8.4	Evaluation . . . . .	98
8.4.1	Dataset . . . . .	98
8.4.2	Experimental Setup . . . . .	99
8.4.3	Results . . . . .	100
8.5	Discussion . . . . .	104
<b>9</b>	<b>Conclusion</b>	<b>105</b>
9.1	Thesis Conclusion . . . . .	105
9.1.1	Content-aware Factorization Model . . . . .	105
9.1.2	Personalized Neural Network . . . . .	106
9.1.3	Object-based Features for Tag Recommendation . . . . .	106
9.1.4	Text-based Features for Tag Recommendation . . . . .	106
9.2	Future Research Direction . . . . .	107
	<b>References</b>	<b>121</b>

## CONTENTS

---



# List of Figures

1.1	An example of tag recommendation for Flickr images. . . . .	4
2.1	A simple MLP with one hidden layer . . . . .	12
2.2	A CNN with one convolutional layer, one subsampling layer and two fully connected layers . . . . .	12
4.1	The distribution of user-image pairs and tags in NUS-WIDE . . . . .	27
4.2	The distribution of users and images in NUS-WIDE . . . . .	27
4.3	The distribution of user-image pairs and tags in Flickr-PTR . . . . .	28
4.4	The distribution of users and images in Flickr-PTR . . . . .	29
5.1	Two people use dissimilar words to assign for two similar images. . . . .	34
5.2	The architecture of CNN-FM and CNN-DenseFM . . . . .	39
5.3	Precision, Recall and F1-Measure for LOPO-2 from rank 1 to 10 . . . . .	46
5.4	Precision, Recall and F1-Measure for LOPO-5 from rank 1 to 10 . . . . .	47
6.1	Characteristics of tags predicted by a personalized content-aware approach . . . . .	50
6.2	The architecture of CNN-PerMLP . . . . .	51
6.3	Comparison of non-personalized and personalized models . . . . .	58
6.4	Comparison of personalized models . . . . .	59
6.5	The results of personalizing images at different levels for NUS-WIDE-1 . . . . .	60
6.6	The results of personalizing images at different levels for Flickr-PTR . . . . .	60
6.7	Feature maps are different based on users. . . . .	61
7.1	When tagging an image, the user is highly influenced by what she sees on the image. In this example, the user chooses “motorcycle” due to its occurrence. Furthermore, the tag “urban” is chosen since the image shows a street, people, cars and other things typical for cities. Our idea is to create an automatic system that uses object detection as a part of it to improve the tag recommendation performance. . . . .	65

## LIST OF FIGURES

---

7.2	The proposed architecture for personalized content-aware tag recommendation. We train one network for the task of image classification and one network for the task of object detection on two different datasets. These networks are finally used to extract image features or detect objects. These features and predictions are used as visual features in order to train a factorization model. . . . .	68
7.3	The architecture of VGG-16 that having 16 weighted layers . . . . .	68
7.4	While the detector cannot recognize any object in the left image, it detects several objects, such as person or boat, in the right image. . . . .	71
7.5	F1-measure and Precision-Recall for NUS-WIDE-1 . . . . .	77
7.6	F1-measure and Precision-Recall for NUS-WIDE-2 . . . . .	78
7.7	For the same object "zebra", while the left image is tagged by "zoo", the "wildlife" tag is assigned for the right image. . . . .	78
8.1	The architecture of a content-aware tag recommendation based on the visual and textual features of images. . . . .	84
8.2	The architecture of VGG-16 that has 16 weighted layers . . . . .	85
8.3	The architecture of VGG-19 that has 19 weighted layers . . . . .	86
8.4	The architecture of ResNet-50 that has 50 weighted layers . . . . .	86
8.5	The simple architectures of the CBOW and Skip-gram network. . . . .	88
8.6	The neural network personalizes the visual-textual features to predict scores of tags. . . . .	94
8.7	The neural network composed of two prediction branches in accordance with different types of image's features. . . . .	95
8.8	The neural network is also composed of two prediction branches in which each element of an image property interacts with other factors to obtain the new representative element. . . . .	96
8.9	The network contains two non-shared image's information branches in which the weights of hidden layers are personalized through two element-wise steps. . . . .	97
8.10	F1-measure and Precision-Recall for FM models rely on VGG-16, Glove, and YOLO extractors. . . . .	101
8.11	F1-measure and Precision-Recall for PITF models rely on VGG-16, Glove, and YOLO extractors. . . . .	102
8.12	Visual and textual features extracted by different extractors boost the performance of FM-based models in different levels. . . . .	102
8.13	Visual and textual features extracted by different extractors boost the performance of PITF-based models in different levels. . . . .	102
8.14	The comparison among factor-based and MLP-based models using the combination of VGG-16 and Glove features. . . . .	103

# List of Tables

4.1	Examples of the Flickr-based dataset . . . . .	26
4.2	Examples of ImageNet instances for the image classification task . . . . .	30
4.3	Examples of COCO instances for the object detection task . . . . .	30
4.4	Characteristics of datasets used to evaluate the performance of the proposed models . . . . .	32
5.1	F1-measure at 5 and 10 . . . . .	48
6.1	The set of 10 most frequent tags in NUS-WIDE-1 . . . . .	57
6.2	The set of 10 most frequent tags in Flickr-PTR . . . . .	57
6.3	Layer characteristics of the convolutional architectures . . . . .	57
6.4	Examples of top 5 recommended tags of CNN-PerMLP, CNN and MP-u . .	61
6.5	Examples having the highest accuracy of top 5 recommended tags . . . . .	62
6.6	Examples having the lowest accuracy of top 5 recommended tags . . . . .	62
7.1	YOLOv2 is a fully convolutional network and is based on the Darknet-19 architecture sketched below. . . . .	70
7.2	Examples top recommended tags of factorization models using different types of features . . . . .	79

## LIST OF TABLES

---

# Chapter 1

## Introductory Material

### Contents

---

1.1	Social Media . . . . .	1
1.2	The Demand for Tag Recommendation . . . . .	2
1.3	Unveiling the Power of Deep Learning . . . . .	4
1.4	Summary of Problems Addressed . . . . .	5
1.5	Main Research Contributions . . . . .	5
1.6	Publications . . . . .	6
1.7	Chapter Overview . . . . .	7

---

### 1.1 Social Media

At the present time, along with the popularity of the Internet, the intended usage of individuals and organizations has also changed dramatically, from searching for information towards socializing. The explosion of Web 2.0 has led to the birth and strong growth of social media services, Internet-based applications, that promote people creating and sharing user-generated contents [59]. Social media services can be categorized as social networking, microblogging, blogging, photo sharing (e.g. Instagram, Flickr), video sharing (e.g. YouTube), and crowd sourcing [61]. In contrast to the traditional media when people stand in a negative position to receive the information from content providers, social media has changed the way people interacting with the resource as well as their communication. Social media has introduced a virtual environment where people take initiative to generate, publish, share and discuss contents. Undoubtedly, a numerous areas of life are significantly impacted by social media, such as health and marketing promotion [23, 101, 155], climate change [152], human relationship [15], digital government [60], human communication and content sharing [12].

Overcoming the limitation of space and time, social media platforms, such as Facebook, Twitter, or Flickr, give people power to connect and interact with anyone at any

## 1. INTRODUCTORY MATERIAL

---

time. The use of these platforms has become widespread with a large number of individuals, organizations, and companies updating their activities on the platforms everyday. Individuals use social media services to share their creative contents, such as photos or videos, give opinions and seek information. In addition to improving the interaction with customers, companies also use the platforms to harness information related to their products or brands from conversations on the Web. With the power ability to spread as well as the innovative generating and managing large-scale data, social media platforms have attracted a growing number of users with enormous and overwhelmingly available information. The fact is exemplified through the following numbers:

- According to [134], around the world in 2016, 2.46 billion people use social media services, accounting for 70 percent of Internet users. They estimated that the number will be up to 2.77 billion in 2019.
- As a market leader, Facebook has around 1.45 billion active users everyday, and they upload around 300 million photos per day [158]. Every minute, users post 510,000 comments and update 293,000 statuses.
- Flickr, a photo-sharing website, has over 90 million active users monthly, and around 75 million users registered as photographers. On a high traffic day, 25 million photos are uploaded to Flickr. Every month, Flickr receives more than 7 billion requests to access and retrieve information via the API kits.

The enormous amount of information has provided great opportunities for organizations to mine and understand their clients in different aspects, such as what they are interested in or what they expect from the companies. Based on the collected knowledge, organizations or companies can optimize the marketing strategy or producing plans to satisfy their current customers, and draw attention of potential consumers to their products. Moreover, social media is a large source of data for researchers to learn precious knowledge about humans, such as their behaviors or emotions. It provides an environment for researches in various fields encompassing computer science, machine learning, social network analysis, data mining, natural language processing, information retrieval, optimization, and mathematics, enabling to evaluate their methodologies or verify their assumption or theories.

### 1.2 The Demand for Tag Recommendation

Undeniably, social media has led the advancement of user-generated contents when it comes to transforming the creating role towards individuals or organizations. However, due to their magnitude and the poor notation, they have negative influences on the accuracy and performance of search engines. In many social media services, an important component, which empowers users to describe and categorize shared resources for their own purposes by using tags, is social tagging, which was introduced to enable searching for resources, such as websites or images. A tag makes reference to a keyword or term assigned to a resource, such as a video or a digital photo, by a tagger. Taggers are not

forced to choose tags from a built-in set of words; instead, they are free to express their perception by using their own words to annotate items. They can be recognized as categorizers or describers [66]. Categorizers tag objects by their own vocabularies for their future search, whereas describers use common words to annotate objects for communities' search.

Summarizing from different works [4, 16, 40, 51, 62, 85, 126, 156], Gupta *et al.* [45] identified several categories of tags based on their performing function including content-based, context-based, attribute, ownership, organizational, purpose, factual, personal, self-referential tags, and tag bundles. For instance, we try to figure out which objects, such as dogs or cats, appear on an image without observing it thanks to assigned tags. With ownership tags, the audience easily identifies who owns the resource. Although the core inspiration of annotating items is to enhance the retrieval process and manage items, a great deal of research has still learned why users tag items in different aspects, which are recapped by Gupta *et al.* [45] in their survey. Obviously, future retrieval is the main reason to tag objects, and taggers can share their opinions with their audiences by using tags; for example, the “nice” word in Flickr is used to express the positive judgment of a user to an image. Moreover, popular tags included in tag clouds may attract users to access others' resources without searching by keywords. Bringing back past memories is also a reason for taggers annotating their resources, such as places or time taken photos. Tagging data are used to improve the performance of many systems, such as information retrieval or item recommendation.

However, tagging is a time-consuming task which discourages users annotating their resources. Users need to be highly concentrated to figure out many possible words in a short moment and type all words in a required space. When typing many words in a short time, users easily make spelling mistakes, or they tend to use meaningless acronyms. Moreover, due to the absence of guidelines, users may use different words for objects of the same user-specific group leading the inaccuracy in the search process. Tag recommendation of social media is an effective system to relieve users from the manual cost of tagging [84, 148]; it will recommend a relevant list of tags, which can describe the content of resources or reflect the favored vocabularies, to a user such that he just needs to click them to select. As described in Figure 1.1, a simple method deployed in the Flickr system is recommending all user's words, which contain the user-provided characters. In addition, with an efficient tag recommendation scheme, the system will appreciably help users avoid incorrect spellings or vague words.

Many approaches have been proposed for tag recommendation based on contents of resources, folksonomies, or contextual data. For the content-based approaches, we learn the mapping between representations of training resources with their associated tags in the training phase, and the model predicts which tag is relevant to a new resource [6, 75]. One example of folksonomy-based approaches is the tag recommendation scheme proposed by Sigurbjörnsson & Van Zwol [130]. For a given image with several user-provided tags, candidate tags are the most related tags with the initial tags with respect to similar scores counting on the co-occurrence data. The external data, such as GPS data, are exploited

## 1. INTRODUCTORY MATERIAL

---

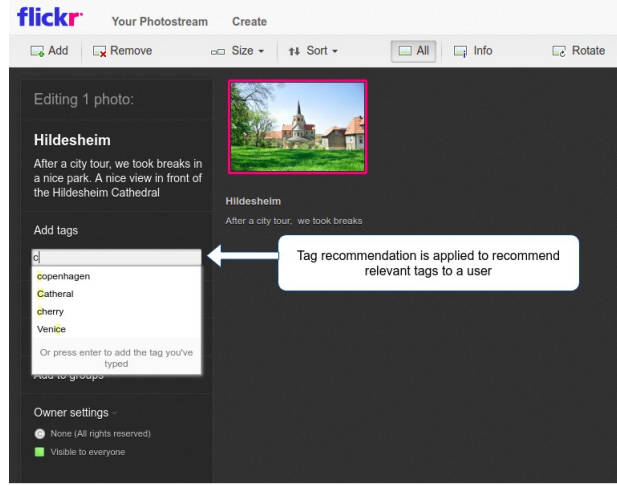


Figure 1.1: An example of tag recommendation for Flickr images.

in the context-based approaches. Chen *et al.* [20] propose a model to extend video's tags. They construct a query with keywords surrounding a given video and use the query to search textual data related to the video; then, the textual content is filtered to obtain keywords suggested for the video.

### 1.3 Unveiling the Power of Deep Learning

The core of deep learning (DL), a machine learning technique developed from artificial neural networks, is the layer concept. Formulated from the layer concept, a DL network is a sequence of stacked layers in which data are passed through these layers. The first layer, known as the input, receives the given data and the output returns useful estimation, such as labels or geographical coordinates of an image. The layers between the input and output, known as hidden layers, transform the given data into more abstract representations.

The first artificial neural networks were invented in 1940s [87] to simulate how neurons in the brain might work to learn or respond to an interaction. In the networks, each node receives data from other nodes, performs a specific computation, and forwards the new data to other nodes. Neurons are gathered in different layers and their connections are known as synapses which weight the transmitted data. Since the first multi-layer perceptron with one hidden layer was introduced, neural networks presently contain dozens, even hundreds of layers with more complex connections.

DL has really leaped forward with the explosion of big available data created a fully conditioned environment in which deep networks having millions of parameters enable to learn their knowledge. However, training DL on CPUs was computationally expensive until GPUs were introduced and has been constantly improved. In addition, numerous studies during the 2000s and early 2010s have accelerated the penetration of DL into various



applications. The most prominent application of DL is in image classification, which seeks to learn labels assigned for images. Deep networks have achieved the best performance in many image datasets, especially in MNIST and CIFAR-10 with nearly 100 percent accuracy [43, 145]. It has also received outstanding achievements in more complicated areas including natural language processing (NLP), object recognition or detection. For example, the DL-based system enables to recognize and localize different objects in images or videos, such as cars, streets, or pedestrians, with high accuracy [113].

## 1.4 Summary of Problems Addressed

Certainly, depending on the social media services, objects of the tag recommendation are diverse, such as images, videos, bookmarks or music tracks, but the focus of this thesis is images. The tag recommendation for images is roughly divided into two categories: non-personalized and personalized models. The non-personalized models suggest the same set of tags for similar images, whereas in the personalized scenarios, different users are recommended different tags to assign for the same image. In this thesis, we seek to learn how to predict user-specific tags for images such that the recommended results reflect both the user tagging history and the image’s content. We are primarily concerned with visual features and two kinds of contextual representations including the image-inside context, which is extracted from images’ pixel values, and the image-outside context, which is obtained from images’ metadata.

Over the next sections, we provide a general overview of machine learning methods, especially about the deep learning and factorization approaches, which serve as the base of the proposed models. By concentrating on the key problems, one of our goals is to analyze how the image content supports the tag recommendation. In addition, we develop a content-aware architecture which takes advantage of deep learning in image-related areas. We discuss some research contributions, publications and provide a chapter overview.

## 1.5 Main Research Contributions

During the research of this dissertation, we have made a number of contributions in tag recommendation. Although many supervised studies for image tag recommendation have been proposed in the last years, there are still several unexploited aspects which need to be investigated to raise the system performance. The main goal is to provide a cohesive view of tag recommendation techniques, identify unsolved problems or unexplored directions, and propose models for tag recommendation. We can summarize the contributions of this dissertation as follows:

- **Content-aware tag recommendation with latent factor models.** We propose using a convolutional neural network (CNN) to implicitly take into account high-level features of images in the first step. Since a user may tend to assign a set of tags to similar images, we propose allowing for the correlations among visual description,

## 1. INTRODUCTORY MATERIAL

---

users, and tags by using factorization machine (FM). This work was published in [96].

- **Personalized visual extracting.** We introduce a personalized layer to fill the gap between the personalized predictor and the multi-label classifier. It is an intermediate layer to transform a visual representation into a personalized form such that the multi-label neural network is utilized to predict multi-tags. We published this contribution in [98].
- **Exploiting image-inside context effects in content-aware tag recommendation.** In addition to visual features, objects appearing in an image can model properties of the entire image. To take into account the object-based factor, the factorization based methods are proposed to receive different features such that predicted tags are capable to represent the correlation among users, visual properties and objects appearing on images. This contribution appeared in [97].
- **Exploiting effects of text based context in content-aware tag recommendation.** We study how effective the textual features are when they are used as a contextual factor in the tag recommendation. Besides the extended factorization based models, we proposed different personalized neural networks which allow predicting different tags for individuals.

The main goal of this thesis is to introduce the deep learning methods for predicting personalized tags for images. Although some of the methods, for example, tensor factorization-based approaches or the deep convolutional networks, are already existing in tag recommendation or multi-label classification, bringing both approaches together to solve content-aware tag recommendation is still a new approach for images.

### 1.6 Publications

The contributions of this thesis were published in several international peer-reviewed conferences. The list of publications are as follows:

1. Nguyen, Hanh T. H. and Wistuba, Martin and Drumond, Lucas Rego and Schmidt-Thieme, Lars (2017). *CNN-FM: Personalized Content-Aware Image Tag Recommendation*. European Conference on Data Analysis (2015), in Archives of Data Science, Series A, ISSN: 2363-9881.
2. Nguyen, Hanh TH and Wistuba, Martin and Grabocka, Josif and Drumond, Lucas Rego and Schmidt-Thieme, Lars (2017). *Personalized Deep Learning for Tag Recommendation*. In Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 186-197).

3. Nguyen, Hanh T. H., Wistuba, Martin, and Schmidt-Thieme, Lars (2017). *Personalized Tag Recommendation for Images Using Deep Transfer Learning*. In Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 705-720).

## 1.7 Chapter Overview

The thesis is organized as follows:

- In **Chapter 2**, we discuss several base methods, which are applied in our proposed models. The main task in this dissertation is classified as supervised learning problems which seek to learn mapping an input to an output from labeled training data. In addition, several techniques to construct the proposed models including deep learning, factorization, and transfer learning are also summarized in this part.
- In **Chapter 3**, we formulate the problem of personalized tag recommendation that we cover and solve in this thesis. We review the problem in several research scenarios: purely personalized tag recommendation, content-aware case, using knowledge from other systems to boost up the tag recommendation and integrating the context-based data into the tag recommendation.
- In **Chapter 4**, we summarize the datasets and present our data-preprocessing procedures in general. The main dataset used in the implementation is the NUS-WIDE dataset, which was collected from Flickr website. We also evaluate the performance of several proposed models in Flickr-PTR, a similar dataset also crawled from Flickr. The source datasets, which provide the “senior” knowledge to the proposed approaches, are also briefly described in this chapter.
- **Chapter 5** investigates a simple but effective method to address the content-aware tag recommendation for images. In this chapter, we develop a hybrid network based on a CNN and a factorization model. With the strong ability to learn visual features, the CNN is responsible to generate a high-level abstract representation of an image, whereas a FM based layer is used to compute scores of tags. The modified pairwise criterion and optimization algorithm are also introduced to learn the model parameters.
- Instead of borrowing the factorization model as the scoring function, in **Chapter 6**, we propose a full neural network to estimate scores of tags. To build the network, we introduce a new layer which captures the correlation between visual representation and users. It receives the output of a CNN-based extractor, performs personalized computations, and passes the new output to a neural network based estimator. We also analyze where the personalized layer will be located to enhance the model’s performance.

## 1. INTRODUCTORY MATERIAL

---

- In **Chapter 7**, we open a new direction in which the inside context data of images are mined to enrich input features. Due to the influence of objects on the user's tagging decision, we argue that objects are considered as a factor improving the predicting capabilities. The object detection is employed to find the object-based description of an image while the base of the visual extractor relies on an image classification network. We extend the models proposed in Chapter 5 and build another factorization-based model to compute tag scores based on the tagging records, visual-, and object-based features.

To increase the performance of classification/object detection components, the transfer learning techniques are employed in which a part or full knowledge of a pre-trained model is passed to the target extractor. We propose putting transfer learning into use so that the running time is mostly used to learn the scoring estimator, and the richer features of images including visual and contextual forms are easily obtained.

- **Chapter 8** continues extending the context-aware scenarios to the outside context of an image, which is hidden under its surrounding text. Given a collection of photos with their description, we aim to address how we can leverage the data to construct a better tag recommendation. In addition to the visual feature generator, word embedding techniques are hired as a textual feature extractor. We evaluate the impact of different context- and visual- features obtained by several extractors to the tag recommendation performance.

Beyond factorization-based models in Chapter 7, several neural network-based architectures are proposed. They inherit the personalized layer from Chapter 6 adding the user information to visual- and textual-features. Individualizing weights of synapses is also taken into account to replace for the user-bias approach such that we expect to improve the performance of the network.

- Finally, we summarize all proposed approaches in **Chapter 9**. We also discuss an outlook in the research area and the future works.

## Chapter 2

# Technical Basics

### Contents

<b>2.1</b>	<b>Supervised Learning . . . . .</b>	<b>9</b>
<b>2.2</b>	<b>Deep Feedforward Network . . . . .</b>	<b>10</b>
<b>2.3</b>	<b>Convolution Neural Network . . . . .</b>	<b>12</b>
<b>2.4</b>	<b>Factorization Model . . . . .</b>	<b>15</b>
<b>2.5</b>	<b>Transfer Learning . . . . .</b>	<b>17</b>

This chapter introduces a set of technical basics that are useful to understand theories and implementation aspects conducted in this thesis. First, we briefly present supervised learning models that support the tag recommendation in general. Next, the core concepts of the proposed models are summarized including deep learning, factorization, and transfer learning. This is not an introduction to the field of machine learning, instead it briefly summarizes several concepts that are often mentioned in this thesis. In order to fully understand the forthcoming concepts, we refer the reader to relevant textbooks [13, 17, 84, 86, 94, 108].

### 2.1 Supervised Learning

Given  $\mathbf{x} \in \mathcal{X}$  is a vector or generally a tensor representation of an item, such as an image or a document, and a target  $y \in \mathcal{Y}$  which is a value assigned to an instance, such as the geographical information of the image or the type of the document. The goal of a supervised learning model is to learn a function  $f$  mapping  $\mathbf{x}$  to  $y$

$$f : \mathcal{X} \rightarrow \mathcal{Y} \quad (2.1)$$

After the function finds the mapping  $f(\mathbf{x})$  for all tuple  $(\mathbf{x}, y) \in \mathcal{D}^{train}$  where  $\mathcal{D}^{train}$  contains all observed records, it will be used to predict the target  $\hat{y}$  for a given unobserved instance  $\mathbf{x} \in \mathcal{D}^{test}$  where  $\mathcal{D}^{test}$  is a set of all unlabeled instances.

## 2. TECHNICAL BASICS

---

In supervised learning, classification and regression, which are distinguished based on the value of outputs, have attracted the most attention of machine learning researchers. If targets are real-valued variables, the problem is called regression. Otherwise, when the response variables of instances are discrete, the problem is known as classification. Classification models seek to find a label  $y \in \mathcal{C}$  assigned to an instance  $x$ , where  $\mathcal{C}$  is the set of all classes with  $|\mathcal{C}|$  being the number of classes. While the classification with  $|\mathcal{C}| = 2$  is known as binary classification, the multi-class classification has  $|\mathcal{C}| > 2$ . If an instance is assigned by a set of target labels  $y \in \{y_i | y_i \in \mathcal{C}\}$ , the classification problem is categorized as the multi-label classification.

In this thesis, we introduce a variant of multi-label classification methods for the tag recommendation problem. The labels of this problem are known as tags which are assigned to an image by a user, and the predicted label sets of a given image are individual for each user. After initializing the parameters at random, a supervised model starts to learn how to predict the set of tags which are close to the target labels on the training set containing observed pairs of instances and targets. The model is evaluated on the test set in which each unobserved instance is assigned by a set of predicted labels. The accuracy of the model is measured by the similarity between the predicted and target label sets.

### 2.2 Deep Feedforward Network

A feedforward neural network, or multi-layer perceptron (MLP), can be seen as a sequence of linear operations followed by a non-linear transformation. The goal of a neural network (NN) seeks to learn an approximate function  $f^*$  which is composed of several different functions  $f^i$  [42]. Each function, formed a *layer*, in the chain is formulated as the concatenation of several logistic regression models, named as *nodes* or *perceptrons*, probably followed by nonlinear transformation. Given an input vector  $x \in \mathbb{R}^m$  and the activation used is a sigmoid function  $\sigma(a) = \frac{1}{1+\exp(-a)}$ . The function  $f^i$  can be formulated as

$$f^i(x) = [\sigma(w_1^T x + b_1), \dots, \sigma(w_n^T x + b_n)] \quad (2.2)$$

where  $w_j \in \mathbb{R}^m$  and  $b_j \in \mathbb{R}$  are the weight and bias values of a model  $j$  contributing to the element  $j$  of the final output. We can redefine Equation 2.2 in terms of a matrix-based formulation as

$$a^i = f^i(x) = \sigma(Wx + b) \quad (2.3)$$

where the set of weights and bias belonging to  $n$  models are grouped in  $W \in \mathbb{R}^{n \times m}$  and  $b \in \mathbb{R}^n$  respectively. Depending on applications of the network, different types of transformation can be applied to the linear combination of the weighted input. Typical

## 2.2 Deep Feedforward Network

activation functions include

$$\text{Sigmoid function: } \text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (2.4)$$

$$\text{Hyperbolic Tangent function: } \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \quad (2.5)$$

$$\text{Rectifier Linear [95]: } \text{ReLU}(x) = \max(0, x) \quad (2.6)$$

$$\text{Leaky Rectifier Linear[83]: } \text{LReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0.01x & \text{otherwise} \end{cases} \quad (2.7)$$

$$\text{Exponential Linear Unit function [24]: } \text{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{if } x \leq 0 \end{cases} \quad (2.8)$$

Another common non-linear operator, which is usually utilized in the output layer for classification, is the softmax function. It is not a single fold function of each input unit and it is defined as

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^C e^{x_j}} \quad (2.9)$$

where  $x_i \in \mathbf{x}$  and  $\mathbf{x} \in \mathbb{R}^C$  are mutually an unit of input and an input vector containing  $C$  elements.

To learn the parameter set  $\Theta$  comprising parameters of all functions, the network is trained with a back-propagation learning algorithm [69, 122]. Let us denote a loss function  $\mathcal{L}$  which defines the difference between the target output  $y$  and the predicted value  $\hat{y}$  generated by the model  $f^*(\mathbf{x}, \Theta)$ . We assume the activation function used in the network is the sigmoid function. The derivation of the back-propagation algorithm begins by applying the chain rule to the partial derivate of the loss function with respect to weights  $W$  of the layer  $i$

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{\partial \mathcal{L}}{\partial f^*} \frac{\partial f^*}{\partial f^{i+1}} \frac{\partial f^{i+1}}{\partial f^i} \frac{\partial f^i}{\partial W} = \delta^i \frac{\partial f^i}{\partial W} = \delta^i \cdot a^i \cdot (1 - a^i) \cdot \mathbf{x}^T \quad (2.10)$$

where  $\mathbf{x}^T$  is the transpose of  $\mathbf{x}$ ;  $a^i$  is the output of the layer  $i$  defined in Equation 2.3. The term  $\delta^i$  called **error** can be defined as

$$\delta^i = \frac{\partial \mathcal{L}}{\partial f^i} = \delta^{i+1} \frac{\partial f^{i+1}}{\partial f^i}$$

Thus, the calculation of the error  $\delta^i$  shown above depends on the values of error terms in the next layer. The errors will proceed backwards from the last layer (the output layer) down to the input layer. After the errors of all layers are known, the gradients of the loss with respect to parameters  $W$  are computed according to Equation 2.10. Figure 2.1 presents a simple architecture of NNs which contains one hidden layer with three neurons and it receives a 4-dimensional vector as an input. In case of 3-class classification, the

## 2. TECHNICAL BASICS

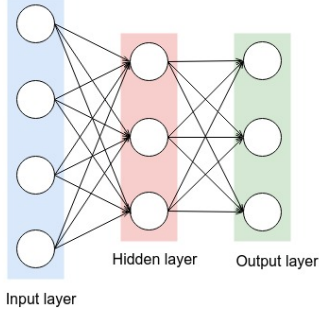


Figure 2.1: A simple MLP with one hidden layer

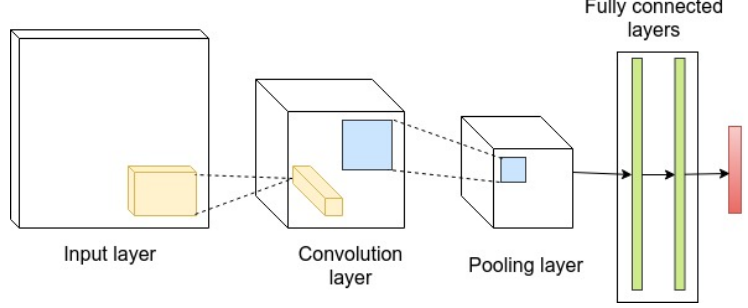


Figure 2.2: A CNN with one convolutional layer, one sub-sampling layer and two fully connected layers

network will compute the probabilities of three classes corresponding to the given input and the class, which has the highest score, will be selected as the final output. The depth of a network can be defined by the number of layers. If a network is deeper than another, it probably has more trainable layers, which contain training parameters, than the previous one.

### 2.3 Convolution Neural Network

A CNN, a specialized kind of NNs, was introduced in [70, 71] and has become rather popular recently. CNNs achieve an empirically good performance in practical applications, especially image classification, recognition, or object detection [68, 131, 138, 145].

Similar to a simple MLP, a CNN is commonly built up by stacking several layers on top of each other generally including convolutional layers, pooling layers, and fully-connected layers. Fully-connected layers are similar to layers of MLPs in which each output is computed by transforming non-linearly the sum of weighted inputs. However, convolutional layers and pooling layers are constructed in a different way, where an output neuron is partly connected with input neurons. In detail, convolutional layers generate an output by combining the input's blocks with weight matrices linearly and passing the combination to a non-linear function. Nevertheless, the combined operator applied in the layers is a convolution operator instead of a dot product. Given a matrix  $I$  and a weight matrix  $K \in \mathbb{R}^{m \times n}$ , known as kernel or filter, the output matrix  $F$  that is generated by convolving  $I$  and  $K$  is known as a feature map. Each pixel of the feature map is the sum of elementwise multiplication between a pixel block of  $I$  and  $K$ . We can formulate the convolved computation as the following

$$F_{xy} = (I * K)_{xy} = \sum_{i=1}^m \sum_{j=1}^n I_{x+i-1, y+j-1} \cdot K_{ij} \quad (2.11)$$

The motivation of using the convolution operator is from the assumption that spatially closer pixels can contribute to the generation of interesting latent features. Moreover, the



sparse connection between inputs and outputs through small kernels, and sharing weights of output functions support to reduce the number of kernel parameters for the network. Compared to normal neural networks in which each output item connects to all input neurons, a convolutional output connects to a small block of input pixels.

In practice, an input of a convolutional layer is a set of matrices, called a tensor,  $I \in \mathbb{R}^{h \times w \times c}$  where  $c$  is the number of channels and  $h, w$  are the size of each channel matrix, known as a feature map. Kernels of the layer become a tensor with 4 dimensions  $K \in \mathbb{R}^{d \times m \times n \times c}$  where  $d$  is the number of feature maps making up the output. Thus, the output of a convolutional layer is also a tensor  $F \in \mathbb{R}^{h' \times w' \times d}$  that includes the convolution results between the input and kernels. If we assume that the stride of kernels is one pixel when they move along the input, the equation 2.11 can be redefined as

$$F_{xyz} = (I * K_z)_{xy} = \sum_{k=1}^c \sum_{i=1}^m \sum_{j=1}^n I_{x+i-1, y+j-1, k} \cdot K_{z, i, j, k} \quad (2.12)$$

Similar to neural networks, an element-wise non-linear transformation is applied to each output unit of feature maps. The most common-used operator is ReLU function which appears in an abundance of CNN architectures for image classification and object detection [50, 68, 78, 113, 131, 137].

In a common CNN, a pooling layer, which is also referred as a downsampling or subsampling layer, goes after a convolutional layer or a block of convolutional layers to reduce the dimension of feature maps and sharpen features. The layer does not contain parameters and relies on an important operator, pooling. The pooling function will replace an output at a specific position with a new value, which can be an average or maximum value of its neighbors. It can be understood in a similar way as a convolution layer when it convolves a filter to the input volume to get subregions of the input feature maps and outputs average or maximum values of these subregions. For example, if the  $\max()$  operation is used to pool values for an output, the operation is named as *max pooling*. When a  $2 \times 2$  filter is applied to pool values for an output and the pooling blocks are not overlapped with the stride  $s = 2$ , it will take the largest value of four input elements in the filtered area.

$$F_{xyz} = \max(I_{2x-1, 2y-1, z}, I_{2x-1, 2y, z}, I_{2x, 2y-1, z}, I_{2x, 2y, z}) \quad (2.13)$$

In contrast, if the average pooling function is applied, the output is the average value of the pooling region.

$$F_{xyz} = \frac{1}{4}(I_{2x-1, 2y-1, z} + I_{2x-1, 2y, z} + I_{2x, 2y-1, z} + I_{2x, 2y, z}) \quad (2.14)$$

A simple CNN with one convolution layer, one max pooling layer, and two fully-connected layers is shown in Figure 2.2. The network receives a squared digital photo having three channels as an input. After going through the convolutional layer and the subsampling layer, a representative vector of the image is created and passed to fully connected layers to estimate the output probabilities.

## 2. TECHNICAL BASICS

---

The back-propagation algorithm is also adapted to carry errors backward from the output to the input layer. If we assume the linear operator is used between different learnable layers, we can illustrate the computation of errors at the layer  $l$  with respect to a loss  $\mathcal{L}$  and errors of the next layer from Equation 2.12 as

$$\delta_{a,b,k}^l = \frac{\partial \mathcal{L}}{\partial I_{a,b,k}} = \sum_{c=1}^d \sum_{i=1}^m \sum_{j=1}^n \delta_{a-i+1,b-j+1,c}^{l+1} \cdot F_{c,i,j,k} \quad (2.15)$$

To avoid overfitting for deep models which contain a huge amount of parameters, several regularization techniques can be applied to learn the models in addition to norm penalty techniques such as Lasso Regression  $L^1$  or Ridge Regression  $L^2$ . While in  $L^2$ , squared magnitudes of parameters are added to the loss function as the penalties, the absolute values of parameters are used in  $L^1$  as the penalties.

A simple method to regularize deep learning models is Dropout [133] which temporarily removes the contribution of several random units in a network during training. At the test phase, the full model with all units is used to predict labels. In the training process, each activation unit of the layer  $l$  is able to be dropped from the network with the probability  $1 - p$  where  $p$  is the probability of a unit being activated.

Given activations  $y^l$  of a layer  $l$ , a switch of an activation is generated with a Bernoulli distribution denoted as

$$m_j \sim \text{Bernoulli}(p)$$

These switches comprised into a vector  $\mathbf{m}^l$ , known as mask, multiply with activations in a element-wise way to produce new outputs which are used as the input in the next layer  $l + 1$ .

$$\begin{aligned} \tilde{y}^l &= \mathbf{m}^l \odot y^l \\ y^{l+1} &= \sigma(\mathbf{W}^{l+1} \tilde{y}^l + \mathbf{b}^{l+1}) \end{aligned}$$

A variant of the dropout, which drops several random connections between the input units and each output unit, is DropConnect method [145] that supports CNNs achieving high performance in classification for MNIST dataset. Given a weight matrix  $\mathbf{W} \in \mathbb{R}^{n \times m}$  of connections between  $m$  outputs of layer  $l$  and  $n$  outputs of next layer  $l + 1$ . The dropped mask  $M^{l+1} \in \{0, 1\}^{n \times m}$  is also generated with Bernoulli distribution. The connection mask is multiplied with the weight matrix to remove inactivate connections. The output of the layer  $l + 1$  using DropConnect during training can be denoted as

$$y^{l+1} = \sigma((\mathbf{W}^{l+1} \odot M^{l+1}) y^l + \mathbf{b}^{l+1})$$

In the testing phase, all output units are computed based on all connections.

Another mechanism to regularize a deep model is increasing the amount of training data by adding augmented instances that are generated from the real data and are labeled with the same value as the real. The technique is called *Dataset Augmentation* which is usually applicable to classification and visual recognition tasks. In terms of images, training data are enriched by adding more transformed images, such as rotated or scaled images, having the same labels as the original training images.

## 2.4 Factorization Model

Factorization approaches, which have attracted a large amount of research in machine learning, demonstrate their powerful capabilities in different applications, especially in the recommendation task. The approaches seek to learn latent features of various entities included in a dataset, such as users and movies in MovieLens dataset [46], or users, resources and tags in Bibsonomy [9] such that the combination of these latent factors is able to reconstruct the target data.

For two-way data, a well-known approach, known as matrix factorization (MF), decomposes a data matrix  $X$  into two representative matrices  $P$  and  $Q$  associated with two categorical entities such that the result of the dot product between  $P$  and  $Q$  approximately reconstructs  $X$  [65]. The process captures the latent interactions between the participating entities existing in the dataset to predict the values of these associations [135].

We use an image rating problem to illustrate the application of MF. All observed relations between the finite user and image sets  $U$  and  $I$  form the data matrix  $X \in \mathbb{R}^{|U| \times |I|}$ . The matrix  $X$  is a sparse matrix in which elements having numerical values is associated with the observed ratings. The target task is to predict the missing ratings of images by users. When applying the matrix factorization method to solve the problem, we seek to learn decomposed matrices  $P \in \mathbb{R}^{|U| \times d}$  and  $Q \in \mathbb{R}^{|I| \times d}$  where  $d$ , the factor dimension, is much less than the number of users and images. The parameter sets  $P$  and  $Q$  are learned based on the observed values such that their product is approximately the original matrix  $X$ .

$$X \approx \hat{X} = P \times Q^T$$

Instead of representing each user's profile and image's profile by  $|I|$ - and  $|U|$ -dimensional rating vectors, the matrix factorization will exploit  $d$  latent features connected with the user and the image where  $d \ll |U|$  and  $d \ll |I|$ . In other words, an image  $i$  is associated with the row  $q_i$  of  $Q$  and each user  $u$  is represented by the  $u$ -th row of  $P$ ,  $p_u$  [65]. The approximate value that the user  $u$  possibly rates the image  $i$  is the result of the dot product between two latent feature vectors denoted as

$$\hat{r}_{u,i} = p_u^T q_i$$

The model's parameters including  $P$  and  $Q$  are optimized in such a way that the reconstruction error is minimal. For example, the Frobenius norm of the difference is used to measure the difference between the constructed and the original matrix. In addition, weighted  $L^2$  regularization terms with respect to different categorized variables are added to avoid overfitting.

$$\mathcal{L}(X; P, Q) = \|X - PQ^T\|_F^2 + \lambda_P \|P\|^2 + \lambda_Q \|Q\|^2$$

where  $\lambda_P$  and  $\lambda_Q$  are the regularization hyper-parameters. A variant of the above factorization model is the non-negative matrix factorization (NMF) approach that forces the

## 2. TECHNICAL BASICS

---

factorized variables being positive. Similar to the MF, NMF factorizes the given matrix into a product of two matrices as

$$X \approx \hat{X} = P \times Q^T$$

where  $P \in \mathbb{R}^{+|U| \times d}$  and  $Q \in \mathbb{R}^{+|I| \times d}$  are non-negative matrices representing for the user and item entities. NMF parameters are also found by solving an optimization problem with a loss defined by the Frobenius norm, or the Kullback-Leibler divergence [72], or the Fast Bregman divergence[76].

For high-dimensional, sparse data which contain more than two participating entities, tensor factorization approaches are attempted to approximate the target tensor factorized into several small  $N$ -order tensors. These approaches have proved their powerful capabilities to deal with different problems such as tag recommendation or signal processing [27, 63, 64, 129, 136]. Two popular tensor generalizations can be examined to decompose a  $N$ -order tensor into a product of multiple low-rank tensors including the canonical/parallel factor (CANDECOMP/PARAFAC) decomposition [47] and the Tucker decomposition/higher-order singular value decomposition (TD) [27, 28, 142].

Tensors known as structures storing the relation of data entities are multi-dimensional arrays with the number of their dimension, called the order, defined by the number of participating objects such as users, movies and times. Let us use a third-order tensor  $\mathcal{A} = (a_{u,i,t} \in \mathbb{R}^{|U| \times |I| \times |T|})$  describing the ternary relation among users, images and tags as the example data. Each triplet  $(u, i, t)$  represents the assigning state having a binary value. If the user  $u$  assigned the tag  $t$  to the image  $i$ , the assignment value is 1. The unobserved assignments are set to 0.

With the TD, the approximation of  $\mathcal{A}$  is found based on three low rank matrices  $\hat{U}$ ,  $\hat{I}$ ,  $\hat{T}$ , where  $\hat{U} \in \mathbb{R}^{|U| \times k_U}$ ,  $\hat{I} \in \mathbb{R}^{|I| \times k_I}$ ,  $\hat{T} \in \mathbb{R}^{|T| \times k_T}$ , and one small tensor  $\hat{C} \in \mathbb{R}^{k_U \times k_I \times k_T}$ , named as the core tensor. Each matrix, known as feature matrix, is estimated to represent an associated entity. In fact, all factorized components contain the model parameters which need to be learned during the training process. We can define the prediction as the multiplication of all components as

$$\hat{a}_{u,i,t} := \sum_{\tilde{u}=1}^{k_U} \sum_{\tilde{i}=1}^{k_I} \sum_{\tilde{t}=1}^{k_T} \hat{c}_{\tilde{u},\tilde{i},\tilde{t}} \cdot \hat{u}_{u,\tilde{u}} \cdot \hat{i}_{i,\tilde{i}} \cdot \hat{t}_{t,\tilde{t}} \quad (2.16)$$

The CANDECOMP/PARAFAC also estimates the tensor  $\mathcal{A}$  by three low rank matrices and one tensor as the TD. However, it assumes that the core tensor is a diagonal which has the value defined as

$$c_{\tilde{u},\tilde{i},\tilde{t}} = \begin{cases} 1, & \text{if } \tilde{u} = \tilde{i} = \tilde{t} \\ 0, & \text{otherwise} \end{cases}$$

The decomposed feature matrices which represent three entities of the tag recommendation have the same dimension  $d$  of the low-rank approximation. In detail, the low-rank matrix of the user entity is defined as  $\hat{U} \in \mathbb{R}^{|U| \times k}$ , the ones of the image is  $\hat{I} \in \mathbb{R}^{|I| \times k}$  and

$\hat{T} \in \mathbb{R}^{|T| \times k}$  is used to express the tag entity. Based on Equation 2.16, the prediction is rewritten as

$$\hat{a}_{u,i,t} := \sum_{f=1}^k \hat{u}_{u,f} \cdot \hat{i}_{i,f} \cdot \hat{t}_{t,f} \quad (2.17)$$

We can find the parameter set by minimizing the square loss showing the difference between the target and estimated values.

$$\operatorname{argmin}_{\Theta} \sum_{(u,i,t)} (a_{u,i,t} - \hat{a}_{u,i,t})^2 \quad (2.18)$$

To avoid overfitting, the penalties of these parameters, such as  $L^2$  terms, are added to the objective function during the training process. For the personalized tag recommendation case, Rendle *et al.* [118, 119] proposed the usage of pairwise losses measuring the magnitude of the difference between the observed and unobserved tags.

In the thesis, we adopt two variants of the factorization models involving factorization machines (FM) [115, 116], which takes advantage of feature engineering technique and the characteristic of the factorization models, and pairwise interaction tensor factorization (PITF) [117], which factorizes the relation tensor into low-rank matrices representing the participating entities but specifies different representations for the tag entity based on its interactions.

## 2.5 Transfer Learning

Transfer learning has recently raised to resolve the difference of features spaces between the training data and the future data [100, 150]. With respect to supervised learning, it aims to use knowledge from a source domain containing high-quality labeled data to improve a predictive function in a related target domain.

As Pan & Yang [100] described in their survey, we can formulate transfer learning in respect of domain  $\mathcal{D}$  and task  $\mathcal{T}$ . While a domain  $\mathcal{D}$  contains a feature space  $\mathcal{X}$  of data instances and a probability distribution of instances  $P(X)$ , a task  $\mathcal{T}$  includes a hypothesis  $f(\cdot)$  learned from the training data and a label space  $\mathcal{Y}$  including all labels. Transfer learning is a process that adapts related information from the source domain  $\mathcal{D}_S$  and its task  $\mathcal{T}_S$  to the target domain  $\mathcal{D}_T$  to improve the predictive function  $f(\cdot)_T$  in the target task  $\mathcal{T}_T$ . The constraint of transfer learning is  $\mathcal{D}_S \neq \mathcal{D}_T$  or  $\mathcal{T}_S \neq \mathcal{T}_T$ .

A transfer learning process can be categorized into a heterogeneous group if  $X_S \neq X_T$ . In contrast, the case where  $X_S = X_T$  is defined as homogeneous transfer learning which we mostly focus in this thesis. In the homogeneous case, certain parts of the source data can be reused with a few target data and Pan & Yang [100] named the type of transfer learning as the instance-transfer approach.

One example algorithm for this problem is *TrAdaBoost*, an extension of the *AdaBoost* algorithm. It assumes that both source and target data have the same feature space and the set of labels but the distributions of data are different. According to it, there is some

## 2. TECHNICAL BASICS

---

source data being useless or harmful to learning the target predictive. It tries to reweight the source domain data to gain the contribution of the “good” source data but reduce the influence of the “bad” data.

Another approach which we adopt in the thesis is the parameter-transfer approach. It assumes that some parameters can be shared between learning models for related tasks. For example, we assume the multi-class image classification is considered as a source task for which a model  $M$  having two parameter sets  $W$  and  $V$  is learn. We aim to find a target model including  $W$  and  $U$  parameters to solve the multi-label classification for images. Because the feature space of two tasks is the RGB color space and two predictive functions have one similar set of parameter,  $W$ , it is possible to transfer the value of  $W$  learned in the source domain to the target task. An advantage of the transfer technique is to save training time for the target model comparing to learn this model from scratch.

## Chapter 3

# Problem Description

### Contents

<b>3.1</b>	<b>Personalized Tag Recommendation . . . . .</b>	<b>19</b>
<b>3.2</b>	<b>Personalized Content-aware Tag Recommendation . . . . .</b>	<b>22</b>
<b>3.3</b>	<b>Personalized Content-aware Tag Recommendation with Transferring Knowledge and Image-based Context . . . . .</b>	<b>23</b>

### 3.1 Personalized Tag Recommendation

Traditionally, tag recommendation learns how to recommend a list of relevant keywords for a given user to annotate a new image. The model can profit from the user's preferences, the resource's content or the context of the tagging task, such as the user's contact, the tagging time or the location of the resource. Simply, we can classify tag recommendation models into two groups: the non-personalized ones, which are chiefly based on the content of resources, and the personalized models depending on the user's historical behaviors. Practically, the recommended tags should reflect the individual characteristics; i.e, various sets of tags are suggested for distinct users to assign the same image. The problem of personalized tag recommendation is to predict a ranked list of tags for a user annotating an image based on the observed tag assignments.

Consider a dataset  $\mathcal{D}$  containing a binary tensor storing associations among users, images, and tags,  $\mathcal{A} \in \{0, 1\}^{|U| \times |I| \times |T|}$  where sets of users, images, tags are denoted  $U$ ,  $I$ ,  $T$  respectively, and  $|U|$ ,  $|I|$  and  $|T|$  are the number of users, images and tags in the dataset. The value of its elements is defined as

$$a_{u,i,t} = \begin{cases} 1, & \text{if the user } u \text{ assigns the tag } t \text{ to the image } i \\ 0, & \text{otherwise} \end{cases} \quad (3.1)$$

The set of assignments  $\mathcal{S} \subset U \times I \times T$  contains all triples reflecting which tags are

### 3. PROBLEM DESCRIPTION

---

assigned to images by users and is defined as

$$\mathcal{S} := \{(u, i, t) | u \in U \wedge i \in I \wedge t \in T \wedge a_{u,i,t} = 1\} \quad (3.2)$$

All pairs of users and images, called posts, are grouped in a set defined as

$$\mathcal{P} := \{(u, i) | \exists t \in T : (u, i, t) \in \mathcal{S}\} \quad (3.3)$$

In addition, the set of tags assigned to an image  $i$  by a given user  $u$  is known as a relevant tag set denoted as

$$T_{u,i}^+ = \{t | t \in T \wedge (u, i, t) \in \mathcal{S}\} \quad (3.4)$$

Otherwise, the irrelevant tag set contains all un-assigned tags of the post  $(u, i)$

$$T_{u,i}^- = \{t | t \in T \wedge (u, i, t) \notin \mathcal{S}\} \quad (3.5)$$

Let us denote some training posts as  $\mathcal{P}^{train}$ , where  $|\mathcal{P}^{train}|$  is the number of training posts, and some training triples as  $\mathcal{S}^{train}$ , where  $|\mathcal{S}^{train}|$  is the number of triples. To solve the personalized tag recommendation, we seek to learn a predictive model with the parameter set  $\Theta$

$$\hat{y}_{u,i} = f(p_{u,i}; \Theta) : U \times I \rightarrow \mathbb{R}^{|T|} \quad (3.6)$$

which computes scores of tags such that the score of a relevant tag  $(\hat{y}_{u,i})_{t^+}$  is larger than the score of any irrelevant tag  $(\hat{y}_{u,i})_{t^-}$ .

For a new post  $p_{u,i} \in \mathcal{P}^{test}$ , the learned model calculates scores of tags and then the tags are sorted in descending order based on these scores. The top- $K$  tags are selected from the ranked list to recommend for the given post.

$$\hat{T}_{u,i}^K := \underset{t \in T, |\hat{T}_{u,i}^K| = K}{\operatorname{argmax}} (\hat{y}_{u,i})_t \quad (3.7)$$

The problem can be reformed as classification formulation with  $\mathcal{X} \in \{0, 1\}^{M \times N}$  encoding posts, where  $N = |U| + |I|$  is the size of the input vector,  $M = |\mathcal{P}|$  is the number of post-based instances and  $\mathcal{Y} \in \{0, 1\}^{|T|}$  encoding the observed tag sets. For example, given a post  $p_{u,i}$  of user  $u$  and image  $i$ , its feature vector  $\mathbf{x}_{u,i} \in \mathcal{X}$  is a binary vector in which elements at the positions associated with the user's and image's index are 1.

$$\mathbf{x}_{u,i} := (\underbrace{0, \dots, \overbrace{1}^u, \dots, 0}_{|U|}, \underbrace{0, \dots, \overbrace{1}^i, \dots, 0}_{|I|}) \quad (3.8)$$

The observed tag set of the post is denoted as  $T_{u,i} = \{t_1, t_2, t_3\}$ , where  $t_\star \in \{1, \dots, |T|\}$  is the index of a tag. The set is shown as a binary vector which contains only  $|T_{u,i}|$  elements at the position  $t_\star$  being 1,

$$\mathbf{y}_{u,i} := (\underbrace{0, \dots, \overbrace{1}^{t_1}, \dots, \overbrace{1}^{t_2}, \dots, \overbrace{1}^{t_3}, \dots, 0}_{|T|}) \quad (3.9)$$



### 3.1 Personalized Tag Recommendation

The scoring function is re-defined as

$$\hat{y}_{u,i} = f(\mathbf{x}_{u,i}; \Theta) : \{0, 1\}^N \rightarrow \mathbb{R}^{|T|} \quad (3.10)$$

To find the parameters  $\Theta$  of the model, we can minimize the loss, such as binary cross-entropy, or maximize the reward, such as the ranking statistic AUC (Area Under the ROC-curve) [26]. For example, if we use the binary cross-entropy loss measuring the difference between the predicted and real tags, the predicted values of tags are converted to 0-1 range. Then, the loss can be presented as

$$\mathcal{L}(y_{u,i}, \hat{y}_{u,i}) = - \sum_{t \in T} (y_{u,i})_t \log(\hat{y}_{u,i})_t - (1 - (y_{u,i})_t) \log(1 - (\hat{y}_{u,i})_t) \quad (3.11)$$

In the pairwise case, to simplify the formulation, we use the relevant and irrelevant tag set,  $T_{u,i}^+$  and  $T_{u,i}^-$ , of the post instead of the binary representation  $y_{u,i}$ . The AUC measure for a post is defined as,

$$\text{AUC}(T_{u,i}^+, T_{u,i}^-, \hat{y}_{u,i}) := \frac{1}{|T_{u,i}^+| \cdot |T_{u,i}^-|} \sum_{t^+ \in T_{u,i}^+} \sum_{t^- \in T_{u,i}^-} H_{0.5}((\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-}) \quad (3.12)$$

where  $H_\alpha$  is

$$H_\alpha := \begin{cases} 0, & x < 0 \\ \alpha, & x = 0 \\ 1, & x > 0 \end{cases} \quad (3.13)$$

Usually, the  $H_\alpha$  is replaced by the sigmoid function defined in Equation 2.4.

In some approaches [115, 117], the training process can be formulated with respect to triples, and the task is slightly similar to regression. A given triple  $s_{u,i,t}$  is encoded into a sparse vector in which elements at the position of  $u, i, t$  are 1.

$$\mathbf{x}_{u,i,t} := (\underbrace{0, \dots, \overset{u}{1}, \dots, 0}_{|U|}, \underbrace{0, \dots, \overset{i}{1}, \dots, 0}_{|I|}, \underbrace{0, \dots, \overset{t}{1}, \dots, 0}_{|T|}) \quad (3.14)$$

Thus, the predictor will map a sparse vector to a real value as

$$(\hat{y}_{u,i})_t = f(\mathbf{x}_{u,i,t}; \Theta) : \mathbb{R}^N \rightarrow \mathbb{R} \quad (3.15)$$

where  $N = |U| + |I| + |T|$ . In the test phase, the model predicts  $|T|$  triples in associated with the given post  $p_{u,i}$ . Similar to the post-based model, top- $K$  tags are selected from the list of tags ranked based on their predicted values.

### 3. PROBLEM DESCRIPTION

---

#### 3.2 Personalized Content-aware Tag Recommendation

While the broad folksonomies, such as del.icio.us, allow many people tagging the same item, the narrow folksonomies, such as Flickr, limit the tagging rights to authorized people [140]. Most resources are annotated by owners who upload and share them. Since an image usually appears in one post, the relationship between it and other factors is not learned if it is only found in the test set. For this reason, the model which has the index-based input described in the above section, will not know its latent information which is used to compute tag scores. This problem can be defined as the item-cold start problem.

To solve the problem, the content of an image is used to replace the one-hot features encoding the image's index. In image datasets, any image carries color information that we can utilize to obtain visual features representing for it. Visual features can be extracted by the hand-crafted techniques, such as color histogram, or machine learned extractors, such as a CNN.

In addition to notations in Section 3.1, we denote the set of digital photos  $\mathcal{R}$  associated with image indices included in the dataset.

$$\mathcal{R} := \{R_i \in \mathbb{R}^{d \times d \times 3} | i \in I\}$$

where  $|\mathcal{R}| = |I|$  is the number of all images. Then, the prediction can be reformulated as

$$\hat{y}_{u,i} = f(p_{u,i}; \Theta) : U \times \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^{|T|} \quad (3.16)$$

The post in this scenarios can be seen as the pair of the  $u$ -th user and the tensor  $R_i$  associated with the  $i$ -th image, and it is expressed as  $p_{u,i} = (u, R_i)$ .

In fact, we tackle the content-aware tag recommendation task based on the relation among users, abstract representations of images, and tags; e.g. given a ternary tag assignment, we firstly aim to extract the visual features of the given image which is an entity of the relation and predict the scores of tags based on the interaction among these features and other entities. We have investigated different architectures to estimate scores of tags and evaluated these models on several publicly available datasets. They are regarded as a complex function built by two sequence models, known as the extractor and the predictor, which are learned simultaneously. The extractor is responsible to map a tensor of a given image to a representative vector

$$z_i = f^e(R_i; \theta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m \quad (3.17)$$

where  $\theta$  is the parameter set of the extraction component, and  $m$  is the size of the feature vector of the image  $i$ . The predictor which calculates scores of tags receives the identification of the given user and the visual features of the image as its input and map it to a score vector.

$$\hat{y}_{u,i} = f^p(u, z_i; \omega) : U \times \mathbb{R}^m \rightarrow \mathbb{R}^{|T|} \quad (3.18)$$

where  $\omega$  is the parameter set of the predictor. The whole of the parameter set  $\Theta$  is the union of  $\theta$  and  $\omega$ . The back-propagation algorithm is employed to transfer errors backward from the predictor to the extractor, and the update of all parameters is executed concurrently.

### 3.3 Personalized Content-aware Tag Recommendation with Transferring Knowledge and Image-based Context

Instead of spending most of the time to learn the extractor, the knowledge of the extractor can be earned from another model. A source model is often established successfully to solve a specific task, such as image classification, that differs from the target task, such as tag recommendation. Later, its extracting skills applied to obtain latent features are passed to the target model to support it in deriving the representations of images. For example, a source model  $g$  for image classification is trained in a source domain  $\mathcal{D}_S$  and it learns how to extract visual features. Then, this skill is used in the tag recommendation to acquire visual features of tagged images. Given source image data  $\mathcal{X} = \{X_i | X_i \in \mathbb{R}^{d \times d \times 3}\}$  and the respective labels  $\mathcal{Y} \in \mathbb{R}^C$ , where  $C$  is the number of classes, the source model  $g$  that is parameterized by two sets  $\theta$  and  $\tau$  predicts the probabilities of labels for a given image

$$g(X_i; \theta, \tau) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^C \quad (3.19)$$

where  $\theta$  is the parameter set of the extractor, and  $\tau$  embraces parameters of the classifier.

We aim to boost the performance of the target model, which estimates scores of tags for a given post, by the source's knowledge. The overlapping component of the source and the target is the visual feature extractor thus the extracting knowledge is practically transferred from the source to the target. Given a target data  $\mathcal{P}$ , as mentioned in the previous section, we seek to learn a model

$$f(u, R_i; \theta, \omega) : U \times \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^{|T|} \quad (3.20)$$

which maps posts to real-valued scores such that for any new post, scores of relevant tags are larger than scores of irrelevant tags. The source extractor's parameters are transferred and frozen during the training process of the target. In other words, when the target is trained on the tag recommendation dataset, the errors are solely propagated back through the predictor's layers and only their parameters are updated.

Additionally, we endeavor to improve the tag recommendation by exploiting different kinds of the image's contexts. The context extractor modeled by the set of parameters  $\vartheta$  seeks to learn a representation of the image context, which is concatenated with other features to compute scores of tags. For instance, we assume the context is realized as a vector  $c_i \in \mathcal{O}$ , which may be extended to a matrix or a tensor, and the vector has  $O$  units. The set of context data associated with dataset images is known as

$$\mathcal{O} := \{c_i \in \mathbb{R}^O | i \in I\}$$

Then, the context extractor is defined as

$$f^o(c_i; \vartheta) : \mathbb{R}^O \rightarrow \mathbb{R}^n$$

where  $n$  is the size of the contextual latent-features. We reformulate the prediction function as

$$\hat{y}_{u,i} = f(u, R_i, c_i; \theta, \omega, \vartheta) : U \times \mathbb{R}^{d \times d \times 3} \times \mathbb{R}^O \rightarrow \mathbb{R}^{|T|} \quad (3.21)$$

### 3. PROBLEM DESCRIPTION

---

As a result, the model parameters  $\Theta$  are the union of  $\theta$ ,  $\vartheta$  and  $\omega$ .

In this thesis, we investigate several architectures to predict tags by using the knowledge transferred from a variety of source models that were originally exploited for different tasks, such as image classification or object detection. Depending on the size of the intersection, the sources contribute some or all of their parameters towards the target model. Further, the context is approximately classified into the inside-image context, which can be revealed through the pixel values of images, and the outside-image context, which can be concealed in the external information source. To achieve reliable representations of the contexts, transfer learning is also adapted in the proposed models, which are being described in details in the later chapters.

## Chapter 4

# Datasets and Pre-Processing

### Contents

<b>4.1 Flickr Dataset</b>	<b>25</b>
4.1.1 NUS-WIDE Dataset	26
4.1.2 Flickr-PTR Dataset	28
<b>4.2 Knowledge-Transfer Dataset</b>	<b>29</b>
<b>4.3 Data Preprocessing</b>	<b>30</b>

In this chapter, we first describe the original datasets that we have selected for experiments. Then, we also present several datasets used to learn the pre-trained knowledge of images and provide extra information. Finally, we show how to filter the original data so that the data can be used for evaluating the proposed techniques.

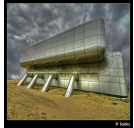

### 4.1 Flickr Dataset

For testing the proposed approaches, we used the Flickr-based datasets which were crawled from the Flickr image-sharing website, [www.flickr.com](http://www.flickr.com). The site is a public storing and sharing service of multimedia which allows users uploading their photos or videos without paying any membership fee [144] for 1-terabyte storage. In this thesis, we focus on solving a problem related to the visual content of digital images so the Flickr's photos are the object that we want to target. When a Flickr user, sometimes called a photographer, associated with an identification number (id) uploads images, he can delineate his resources by titles and descriptions. Moreover, only he or authorized users, who are given permission by him, are able to assign keywords, called tags, which support for image retrieval or organization. In addition to human tags, the Flickr system annotates uploaded photos with its predicted words, named as machine tags. Users can decide to publish these photos to be viewed by anyone or keep their images private and viewable by themselves. Varied licensing parameters can be customized for images, such as All Rights Reserved or Creative Commons.

## 4. DATASETS AND PRE-PROCESSING

Aside from the original resolution, Flickr provides different sizes of photos, such as thumbnail or square photos. Furthermore, we can work with Flickr to find and retrieve information of a photo and its owner by using Flickr API<sup>1</sup>. For example, we can know who has commented on a photo, its geographical data or all tags that a user has used [74]. Depending on the application, various information can be extracted from the Flickr archive. In this thesis, we aim to find the solution for personalized tag recommendation that works with relations of three entities encompassing users, images, and tags, and the image content including visual and textual features. For this reason, we only derive the information of taggers, tags and digital images with their descriptions and titles from Flickr based on the list of photo’s ids involved in two datasets, NUS-WIDE and Flickr-PTR. Examples of the dataset instances are shown in Table 4.1.

Table 4.1: Examples of the Flickr-based dataset

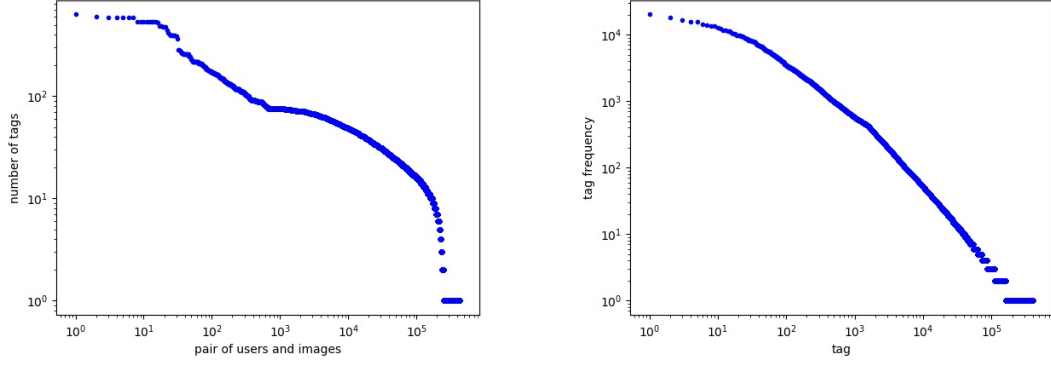
User	Image	Tag	Title	Description
88585855@N00	 125112126	dynamic, paris, range, metro, france, high	Waternet Amsterdam	The aliens have landed! They choose Amsterdam, wonder why? Shot with multiple exposures merged to HDR. 3 exposures. This is actually an existing building and no render (boost-ergemaal spaklerweg)
95681724@N00	 2430302796	subway, train, brooklyn		Waiting for a train back into Manhattan after dinner at Dumont in Brooklyn.

### 4.1.1 NUS-WIDE Dataset

The NUS-WIDE dataset [22] can be viewed as the set of multi-label classification data in which an image is labeled by one or more tags. The authors used Flickr API to crawl more than 300,000 images and removed images having inappropriate size. After removing tags appearing less than 100 times, the authors continued discarding noise tags, which contain spelling errors, names or meaningless words, by mapping them against WordNet[92]. The total of remaining unique tags are 5,081 tags which have the semantic correlation. In addition, the authors also provided a collection of low-level features including color histogram, color moments and bag of visual words which are detailed in [22].

However, the dataset does not include the relations among users, images and tags which are crucial to perform the tag recommendation experiments. Therefore, we crawled color images with their tags and taggers based on their ids recorded in the NUS-WIDE

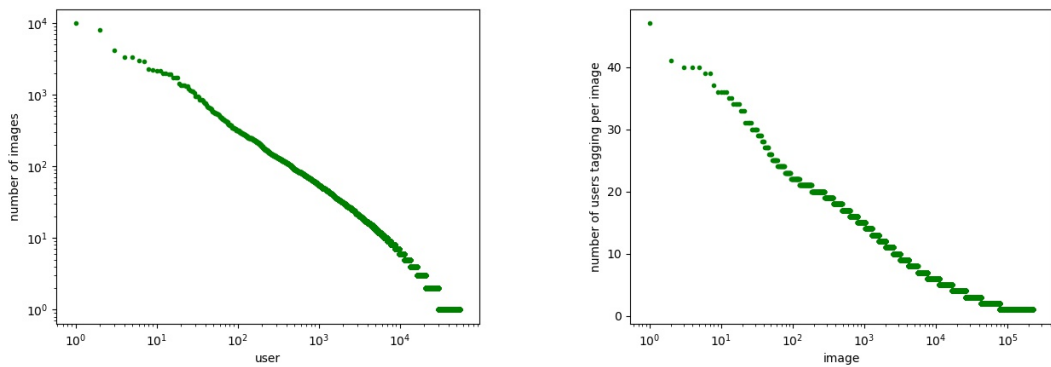
<sup>1</sup><https://www.flickr.com/services/api/>



(a) The number of tags per pair of users and images      (b) The distribution of tags with respect to posts

Figure 4.1: The distribution of user-image pairs and tags in NUS-WIDE

dataset. Figure 4.1a illuminates the distribution of tags per pair of users and images while Figure 4.1b exemplifies the frequencies of tags assigned per post. Through these figures, we can recognize that most user-image pairs consist of less than 20 tags including noise words, and around 90 percent of tags appear less than 10 times in the entire dataset. The distribution of users and images is described in Figure 4.2. As displayed in Figure 4.2a, a user annotates around 10 images with unique tags, whilst the number of users assigning tags for an image is limited due to the tagging authority as seen in Figure 4.2b. In fact, more than 50 percent of images are annotated only by their owners who share them on Flickr site.



(a) The distribution of images per user      (b) The number of users tagging per image

Figure 4.2: The distribution of users and images in NUS-WIDE

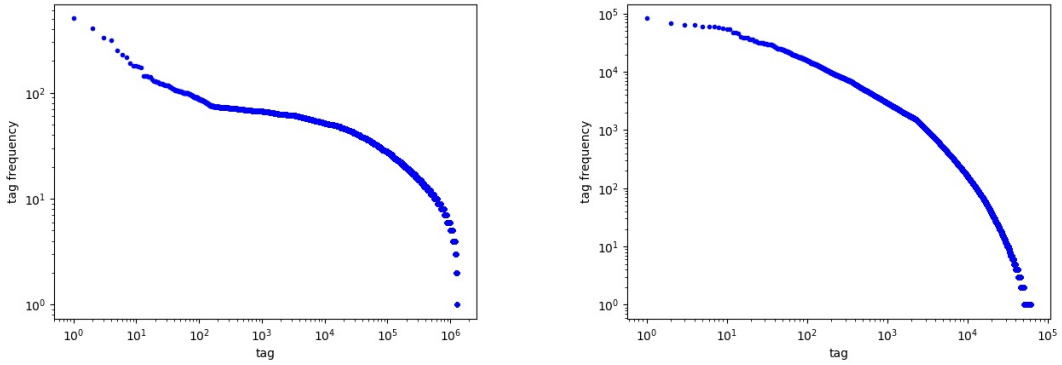
According to the distribution of this dataset, we can see that the data are really sparse,

## 4. DATASETS AND PRE-PROCESSING

---

and they contain rarely used tags and an abundance of users occurring in few times. To conduct the experiments, we filtered the dataset through several steps including keeping English tags and removing inappropriate users, images and tags so that the proposed models have enough training samples to be learned. The general filtering steps are being described in Section 4.3.

### 4.1.2 Flickr-PTR Dataset



(a) The number of tags per pair of users and images      (b) The distribution of tags with respect to posts

Figure 4.3: The distribution of user-image pairs and tags in Flickr-PTR

The other dataset where we carried out the experiments for several proposed approaches is Flickr-PTR (Flickr-Photo Tag Recommendation) published in [88]. The authors provided two datasets for different tasks: automatic image annotation (AIA) and photo tag recommendation (PTR). These datasets were accomplished in a similar way to NUS-WIDE in which images were crawled from Flickr through the Flickr API. However, they did not drag randomly, but they searched and retrieved images based on 2000 popular nouns of WordNet. From the top 2000 images of each searching category, they also eliminated images, which did not satisfy several requirements, such as not containing the creative common license or tags. Instead of removing noisy tags, such as tags describing camera meta-data or Flickr awards, like Flickr-AIA, the authors skipped the cleaning tags process when they produced Flickr-PTR. They collected around 2 million images in total with a set of one million tags in which there are a large number of noisy words. The details of the original dataset are reported in [88].

For the reason that the proposed approaches require using color images which are not enclosed in the published dataset, we crawled again these images based on their ids through Flickr API. Initially, we retrieved 1,263,836 images with 62,267 tags used by 211,015 users, and the total of user-image pairs considered as instances used for tag recommendation are 1,291,717 pairs. The distribution of tags over all user-image pairs is displayed in Figure



4.3a, and the frequencies of tags assigned to each pair are illustrated in 4.3b. In addition, the distribution of users and images shown in Figure 4.4 indicates that most images are annotated by few users and a large number of users assign tags less than 10 times. Similar to the NUS-WIDE case, several pre-processing stages were taken to remove the noisy instances and generate the useful data used for the model evaluation.

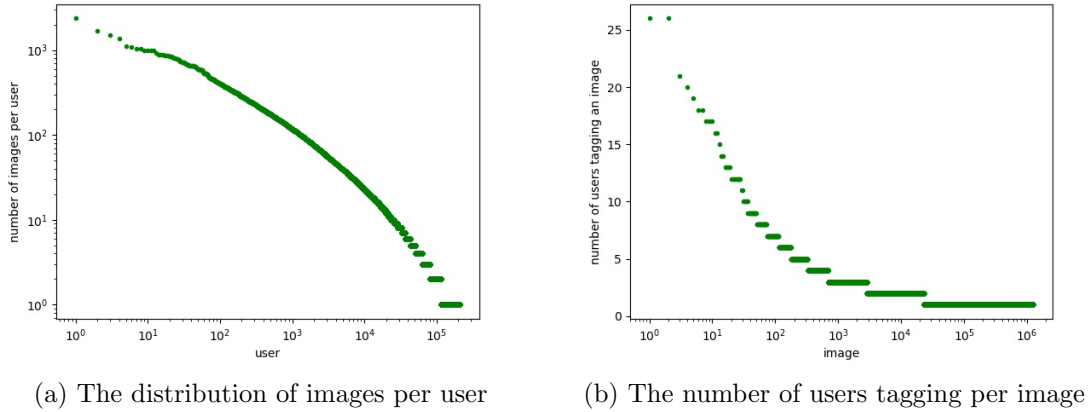


Figure 4.4: The distribution of users and images in Flickr-PTR

## 4.2 Knowledge-Transfer Dataset

To improve the quality of feature extractors in the proposed models, the transfer learning technique was adopted to pass the feature-representation knowledge from source datasets to tag recommendation datasets. We have worked with two image datasets which act as the sources providing knowledge for the extractor of tag recommendation models.

### ImageNet

The image classification dataset is one of three main datasets designed for an annual challenge of object category classification and detection, known as *ImageNet Large Scale Visual Recognition Challenge*. The construction of ImageNet is based on the semantic WordNet hierarchy in which images are grouped in each node of the tree presenting a set of synonyms [31]. Each image is labeled by one of 1000 categories which are selected from all synsets such that these synsets do not overlap with each other. There are approximately 1.2 million train images, 50 thousand validation images, and 100 thousand test images. The details of the collection and the cleaning process are provided in [123]. Table 4.2 shows several examples of the ImageNet dataset in which each image is identified with a single label, such as *tiger* or *hog*.

## 4. DATASETS AND PRE-PROCESSING

---

Table 4.2: Examples of ImageNet instances for the image classification task





Image	Class
	tiger
	hog

Table 4.3: Examples of COCO instances for the object detection task

Image	Bounding Box	Category
	473.07	dog
	395.93	
	38.65	
	28.67	
	204.01	person
	235.08	
	60.84	
	177.36	

### Microsoft COCO

The dataset is designed to illustrate the object recognition problem. It comprises around 91 object categories labeled for more than 2.5 million labeled instances in 328 thousand images [79]. Depending on the version of the dataset, the categories of labeled objects can be 91 or 80, and the splitting protocol is approximately 50 percent images located in the training set, 25 percent for the validation set and 25 percent for the test set. We worked with the 2014's subset that contains 82,783 training images annotated with 80 objects. Examples of the COCO dataset in Table 4.3 show that each object detected in an image is put into a specific category, such as *dog*, and its location is illustrated by a bounding box with four coordinate values.

### 4.3 Data Preprocessing

As mentioned in Section 4.1.1 and 4.1.2, we crawled images according to the identification numbers of images given by the original datasets. We gathered not only the information of tags and taggers but also color photos and their text data including their descriptions and titles. Because we focused on the content and context of digital images comprising visual and textual information, we ignored another contextual data, such as time, geographical data or the meta-data of users, such as their followers, contacts or groups where they belong to. Thanks to Flickr API allowing users to retrieve color images with different sizes in addition to the original size, we utilized small resolution images with the square dimension of 75 pixels for experiments in Chapter 5 and large square images with the 150-pixel dimension in Chapter 6 while medium resolution images were used in Chapter 7 and 8. Additionally, to increase the number of image instances for training processes, we generated four alternatives including scale, top-left, center and bottom-right patches for

each image in Chapter 6. Indeed, the size of each NUS-WIDE patch was  $75 \times 75$  pixels, and each Flickr-PTR patch was  $55 \times 55$ . Since we intend to evaluate the effectiveness of supervised features against hand-crafted features, we culled several subsets of low-level features, such as color histogram, from the hand-crafted feature collection provided by the NUS-WIDE authors. However, we obtained color histogram features of Flickr-PTR images by ourselves because they are not supplied by the dataset’s authors.

Compared to the visual information of an image, textual data can be viewed as a string of characters reflecting the additional content and context knowledge of an image, such as time or location. In the raw form, a textual representation belonging to an image corresponds to a sequence of characters, which can contain some meaningless words, such as misspelled words or hyperlinks, and noisy words, such as camera meta-data or overused punctuation marks. We cleaned texts retrieved from Flickr by removing hyperlinks, blocks of punctuation, stop-words, names and replacing emoticons with the associated words. After those steps, there are around 90 percent of images involving textual information to be used for generating vector-space representation later.

To remove the noisy tags, we relied on the most popular tags provided by the dataset authors or we mapped received tags against WordNet. The 1000 and 100 most popular tags were applied in Chapter 5 and in Chapter 6 respectively while in Chapter 7 and 8, by matching with WordNet, only English tags were kept. Afterward, we eliminated images having no tags and users who did not annotate any image. Further, because we are going to use the index-based values, such as the user’s and tag’s indices, in the tag recommendation models, the datasets should be encoded in terms of values of indices and all of them are started from zero. The proposed models are easily able to convert users’ and items’ indices to one-hot vectors, which contain zero for all elements except the index position, to use as feature vectors during training and testing processes.

Since the datasets are very sparse, we concentrated to evaluate the performance of the proposed approaches in their dense parts. To retrieve the useful data for experiments, we applied the  $p$ -core at level  $k$  protocol to filter datasets [56] such that each user, tag and image occur at least  $k$  times. Since most images are assigned by few users, we did not apply the  $p$ -core filter at the same level for all entities. Instead, the  $p$ -core at level  $k_1$  and  $k_2$  were employed for users and tags so that each user annotates at least  $k_1$  images, and each tag is used at least  $k_2$  times. In addition, there are many tags which dominate all images of a user and do not illustrate the specific characteristic of the user. For this reason, we filtered tags which are assigned to more than 50 percent images by a user. To assess the performance of the proposed models, we used a variant of the evaluation methodology, called *LeavePostOut* or *leave-one-post-out*, described in [57]. Instead of picking randomly one image of each user to put into the test set, we picked more than one image such that instances of the test set are 10 to 20 percent of the whole dataset.

Depending on the filter levels, we generated four subsets of the NUS-WIDE dataset, named as LOPO, NUS-WIDE-1, NUS-WIDE-2, and NUS-WIDE-3, while one subset of the Flickr-PTR was created for the evaluation. In **LOPO**, the data refining process relied on the  $p$ -core protocol at the same level  $k = 2$  for all entities, and the dominant tags were

#### 4. DATASETS AND PRE-PROCESSING

---

Table 4.4: Characteristics of datasets used to evaluate the performance of the proposed models

Characteristics	LOPO	NUS-WIDE-1	NUS-WIDE-2	NUS-WIDE-3	Flickr-PTR
Users $ U $	3,009	1,000	1,999	4,476	323
Images $ I $	13,334	27,662	90,483	122,958	29,095
Tags $ T $	983	100	1,661	2,078	133
Triples $ S $	111,407	81,263	634,739	858,762	94,387
Posts $ P $	29,089	27,858	95,130	129,223	29,096
Training posts $ P^{train} $	26,048 ( <i>LOPO-2</i> ) 27,946 ( <i>LOPO-5</i> )	25,858	76,842	101,622	23,402
Test posts $ P^{test} $	3,009 ( <i>LOPO-2</i> ) 1,143 ( <i>LOPO-5</i> )	2,000	18,288	27,601	5,694

still kept. Two variants of LOPO, which are known as **LOPO-2** and **LOPO-5**, were made based on how we split the train/test set; the original protocol *LeavePostOut* was applied to create LOPO-2 while only images of users having at least five posts in LOPO-5 were considered during the division process. To produce the NUS-WIDE- $\star$  sets, we utilized the  $p$ -core at different levels for users and tags as well as eliminated the dominant tags. The filter levels  $k_1 = 20$  for users and  $k_2 = 100$  for tags were used to produce **NUS-WIDE-2** while  $k_1 = 10$  and  $k_2 = 100$  were found in the **NUS-WIDE-3** generation process. Moreover, the adapted *LeavePostOut* protocol was deployed for the train/test division; for each user, 20 percent of posts were selected for the test set. For **NUS-WIDE-1**, we sampled 1000 users before refining to get a 10-core dataset referring to users and tags whereas we applied  $k_1 = 40$  and  $k_2 = 400$  to filter **Flickr-PTR** and then picked 500 users with their associated images and tags to get the final dataset. For each NUS-WIDE-1 user, two posts were randomly selected to put into the test set, and 20 percent of Flickr-PTR posts for each user were comprised in the test set. These subdivided datasets can be described with respect to users, images, tags, triples and posts as in Table 4.4.

## Chapter 5

# Factorization Content-Aware Tag Recommendation

### Contents

---

<b>5.1</b>	<b>Introduction . . . . .</b>	<b>33</b>
<b>5.2</b>	<b>Review of Tag Recommendation . . . . .</b>	<b>34</b>
<b>5.3</b>	<b>Problem Extension . . . . .</b>	<b>36</b>
<b>5.4</b>	<b>Content-Aware Factorization Model for Tag Recommendation</b>	<b>37</b>
5.4.1	Factorization Machines . . . . .	37
5.4.2	Convolutional neural networks . . . . .	38
5.4.3	Content-aware Factorization Machines . . . . .	39
5.4.4	Optimization . . . . .	41
<b>5.5</b>	<b>Evaluation . . . . .</b>	<b>43</b>
5.5.1	Dataset . . . . .	44
5.5.2	Experimental Setup . . . . .	44
5.5.3	Results . . . . .	45
<b>5.6</b>	<b>Discussion . . . . .</b>	<b>47</b>

---

### 5.1 Introduction

As already discussed in Chapter 1, the explosion of social networks has been connected with the proliferation of digital resources created and shared by users of these networks. Tag recommendation systems enable users to escape from a time-consuming task, which is thinking of tags for their resources. These systems suggest a set of relevant tags to the user and can be dependent or independent on the user's tagging behaviors. In an independent scenario, the systems are founded on the contents of resources, such as objects appearing in images, to predict candidate tags for given images. On the contrary, in a dependent

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

---

situation, they consider the user’s historical behaviors to estimate a specific set of tags for a given user to annotate a new image.

Actually, different taggers prefer to use their words to describe images in their own ways. An example of the fact is shown in Figure 5.1. For this reason, it is practical to



(a) The user *10431071@N02* annotates the image with tags: *cats*, *cat*, and *kitten*.



(b) The user *10604163@N00* annotates the image with tags: *animal*, and *kitty*.

Figure 5.1: Two people use dissimilar words to assign for two similar images.

recommend a personalized list containing the user’s favorite tags to annotate a new image. In the Flickr domain, where we want to study the tagging task, the social tagging system allows users to tag their friends’ photos, but small subsets of 58 million observed tags were used by another except owners of images [85]. Therefore, the system usually has to anticipate relevant tags for new images, which are not learned by the model. Instead of using the index-based relations, we look for a substitute connection among three entities of tag recommendation so that for any new image, the similar relation is already discovered by the system. Along with that, users often annotate their images with vocabularies relating to the contents or contexts of images [130], hence the visual information is likely considered as an important factor that enables to improve the performance of tag recommendation, especially for unobserved images. If recommended tags illustrate both the characteristics of users and images, they are more relevant to a given user-image pair.

From this motivation, we propose a personalized tag recommendation approach using the visual content of images and the users’ historical tagging information to predict an applicable tag set for a given image. Through the performance of the proposed model, we prove the capability of image features in enhancing the recommendation performance. Unlike several state-of-the-art approaches which ignore visual features or use low-level image representations, our approach firstly extracts visual features of images with a CNN. Then, these features are fed to the factorization model to predict a ranked list of tags that is built upon pairs of users and images.

### 5.2 Review of Tag Recommendation

A vast amount of studies have been launched to explore various aspects of tag recommendation. Recommending tags in accordance with the content of images is the most natural approach; however, the selected tags are not solely based on the image but also

depend on the user. A simple approach proposed by Abbasi *et al.* [1] is a cluster-based model which groups images in different clusters. Each cluster is associated with a set of characteristic tags which are achieved by ranking tags based on user frequencies. A new image is assigned to the closest cluster, and the top representative tags of this cluster are recommended to the image. Chen *et al.* [18] presented a system named *SheepDog* using a Support Vector Machine (SVM) to learn the confidence of concepts which an image can belong to. In the inference phase, the top-ranked concepts are extracted for a new image. These concepts are used as keywords to search related groups and tags to recommend for this image. Another approach offered by Lee *et al.* [73] is to recommend tags on the basis of posterior probabilities that are computed according to the similarity between images in terms of their visual features. In [77], the authors introduced a neighbor voting algorithm to accumulate relevant scores from the votes of similar images. For a given image annotated by a user, several nearest neighbors which are visually similar to the target image are selected, and only images of other users can contribute their tags conjoined with voting values.

Several researchers have chosen to exploit user-provided tags as their direction. Generally, a set of initial tags is required at the beginning of the predicting process, and candidate tags for each provided tag are specified by the correlation measures. These candidates are aggregated to bring forth the final list for the image. A clear example of this direction is the collective knowledge approach proposed by Sigurbjörnsson & Van Zwol [130]. They assembled candidate tags for each user-provided tag based upon the concurrent occurrence of these tags and the initial one in the same image. After the candidate list was generated, they employed an aggregation strategy, voting or summing, to filter the list. Finally, the promotion scheme, which weighs the reliability of recommended tags in keeping with the user-defined tags, was applied to select the top tags counted on their final scores. Garg & Weber [35] put forth a personalized approach that combines the individual suggestion and the global tag co-occurrence to narrow down recommendations for a given user. Wu *et al.* [154] propounded an approach depending on three methods to measure the correlation of tags: (1) tag co-occurrence computed in the similar way to [130], (2) tag content correlation estimating the similarity between tags based on the visual information of image sets marked with tags and (3) image conditioned tag correlation measuring the similarity of tags in regard to the target image.

Several approaches combine both content-based and user-provided tag techniques to predict the promising tags. An example of these methods is the model introduced by Sevil *et al.* [127]. First, they retrieved several images, which contained the initial tags, and collected tags from these images to form a unique tag set. Next, the candidate tags were selected from the set according to their weights, which were computed based on the visual-based similarity of the target image and the retrieved images.

The meta-data of images have attracted the attention of many researchers. Tagging photos using users' vocabularies [106] is an approach in which the candidate tags of a given image are collected from its neighbors appearing in the user's tagging history. The neighbor retrieval protocol relies on the similarity between the query image and the database im-

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

---

ages according to GPS, time and visual features. The model will choose the most frequent tags from the list to recommend to the user. Rae *et al.* [107] integrated the information extracted from personalized and collective contexts including the user’s own photos, photos retrieved from his contacts or from his groups, and all photos of the system. The recommendations generated for each context were consolidated in a final recommended set of tags.

Factorization models applied for tag recommendation show a good performance by exploiting the relation among users, images, and tags to find the interaction weights for a new instance. One of the state-of-the-art approaches is Pairwise Interaction Tensor Factorization (PITF) [117] that models all pairwise interactions of three entities. Factorization Machine (FM) proposed by Rendle [115] takes advantage of the feature engineering flexibility and strong prediction capability of factorization to suggest a ranked list of tags for a user.

Multi-label image classification can be seen as the content-based tag recommendation which predicts labels associated with an image. For this reason, we are able to adapt these models for the tag recommendation problem. The CNNs were reconstructed in [41, 149] to predict candidate labels for a given image. They learned parameters of the predictor based on pairwise or Weighted Approximate Ranking (WARP) loss functions [41] or predicted labels from arbitrary trained objects [149]. However, these kinds of methods do not consider personalization when suggesting labels for images.

In our research, we aim to learn a personalized tag recommendation that relies on the tagging history, like factorization models, and the content of images, like multi-label classification models. The proposed approach integrates the strength of factorization approaches, which successfully capture the personalized tags, and the advantage of deep learning models, which predict tags connected to visual contents.

### 5.3 Problem Extension

As we have already mentioned in Chapter 3, the problem that we want to address is to predict a set of the user’s tags revealing his preferences. Instead of simply exploiting the relation of users, images and tags based on their identification numbers, the RGB information of images is adapted to derive the representation of images. Then, the image features replace the image’s index in the relation to compute scores of tags.

Each image  $i$  is represented by a tensor  $R_i$  containing pixel values of the image. The collection of all tensors is defined as  $\mathcal{R} := \{R_i | R_i \in \mathbb{R}^{d \times d \times 3} \wedge i \in I\}$  where  $d$  is the dimension of the given square image. With respect to a given user and a digital image, a post  $p_{u,i} = (u, i)$  can be re-written in terms of a user  $u$  and a RGB image  $R_i$  as  $p_{u,i} = (u, R_i)$ . The two terms are used interchangeable in this section.

Given a training post  $p_{u,i}$  and its observed tags  $T_{u,i}$  represented by a binary vector  $y_{u,i} \in \{0, 1\}^{|T|}$ , we seek to learn a model  $f$  defined as

$$\hat{y}_{u,i} = f(u, R_i; \Theta) : U \times \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^{|T|} \quad (5.1)$$



## 5.4 Content-Aware Factorization Model for Tag Recommendation

where  $\Theta$  is the parameter set of the proposed model. The model guarantees that for any pair  $(u, R_i)$ , the difference between scores of relevant tags  $(\hat{y}_{u,i})_{t+}$  and irrelevant tags  $(\hat{y}_{u,i})_{t-}$  is maximized. We can decompose  $f$  into two sub-functions  $f^e$  and  $f^p$

$$f(u, R_i; \Theta) = f^p(u, f^e(R_i; \theta); \omega) \quad (5.2)$$

where  $f^e$  is the extractor that derives the visual representation of the image  $R_i$ , and  $f^p$  computes the tag scores based on values of  $f^e$  and  $u$ . Additionally,  $\theta$  and  $\omega$  are the parameter sets of the extractor and the predictor respectively and are also the subsets of the model parameters  $\Theta$ .

By splitting this way, it is easy to see that our goal is to find both the extractor  $f^e : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m$ , which provides a  $m$ -dimensional feature vector describing the content of the given image, and the predictor  $f^p : U \times \mathbb{R}^m \rightarrow \mathbb{R}^{|T|}$ , which computes scores of tags.

## 5.4 Content-Aware Factorization Model for Tag Recommendation

### 5.4.1 Factorization Machines

The advantages of factorization models and feature engineering are incorporated in a FM model, which can be successfully applied to solve classification, regression or ranking problems [115, 116], especially for personalized tag recommendation. In the case of tag recommendation, an input of a FM model, which is entirely established on triadic relationships of users, images and tags, is a sparse vector  $\mathbf{x} \in \mathbb{R}^N$  characterizing the triplet  $(u, i, t)$  where  $N = |U| + |I| + |T|$ .

$$\mathbf{x}_{u,i,t} = \left( \underbrace{0, \dots, \overset{u}{1}, \dots, 0}_{|U|}, \underbrace{0, \dots, \overset{i}{1}, \dots, 0}_{|I|}, \underbrace{0, \dots, \overset{t}{1}, \dots, 0}_{|T|} \right) \quad (5.3)$$

The scoring function of the FM model is generally formulated as:

$$(\hat{y}_{u,i})_t = f(\mathbf{x}_{u,i,t}) = w_0 + \sum_{j=1}^N x_j w_j + \sum_{j=1}^{N-1} \sum_{j'=j+1}^N x_j x_{j'} \sum_{q=1}^p v_{j,q} v_{j',q} \quad (5.4)$$

where  $w_0 \in \mathbb{R}$  is the global bias,  $w \in \mathbb{R}^N$  is the weights of input features, and  $\langle \mathbf{v}_j, \mathbf{v}_{j'} \rangle$  is the interaction between the  $j$ -th and  $j'$ -th variables. Each interaction variable  $\mathbf{v}_j$  is a  $p$ -dimensional vector and can be interpreted as a row of an embedding matrix  $V \in \mathbb{R}^{N \times p}$  where  $p$  is the number of factors.

With the configuration of the input, the FM is limited to solve the tag recommendation which is purely dependent on the historical tag assignments of users and images. However, owing to the strength to encrypt diverse kinds of features, the FM is a pragmatic approach to be applied for the content-aware tag recommendation. To predict tag scores counting

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

---

on both the relations between different entities and the content of images, a hand-crafted or trainable extractor should be included in the recommendation model to produce the visual representation of images. The FM will act as a prediction layer, which receives the extracted image features and computes the score of a tag based on the interaction between these features, the tags, and the given user.

### 5.4.2 Convolutional neural networks

As mentioned in Chapter 2, a CNN is a special kind of neural networks, which enables to represent high-level abstractions of images. It has demonstrated the strong capability to classify images in multiclass datasets, such as MNIST [145], CIFAR-10/100 [78] or ImageNet [68]. Ordinarily, the model involves one or more convolutional layers generating multiple 2-dimensional feature maps by sliding learnable filters across the input volume. A sub-sampling layer, which comes after a convolutional layer, pools a rectangular block of pixels in the previous feature maps to produce an output. After several convolutional and subsampling layers, a few fully-connected layers follow the last convolutional or pooling layer to identify a relevant label for the input image.

In a classification case with  $C$  labels, the model can be defined as

$$f(X; \Omega) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^C$$

where  $X$  is an image instance and  $\Omega$  contains parameters of the CNN. The function is probably seen as a chain of the extractor  $f^e$ , and the predictor  $f^p$  as we described in the previous section where

$$\begin{aligned} z_i &= f^e(X_i; \theta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m \\ \hat{y}_i &= f^p(z_i; \kappa) : \mathbb{R}^m \rightarrow \mathbb{R}^C \end{aligned}$$

In these functions,  $\theta$  and  $\kappa$  are parameters that need to be learned simultaneously during the training process. The extractor is a sequence of convolutional and subsampling layers, sometimes called a convolutional block, and the predictor or classifier is a merger of fully-connected layers.

Compared to the problem we intend to solve, the predictor is the different part between the classification model and the tag recommendation model. While the user information is not manipulated to estimate labels' probabilities in image classification, the tag recommendation approach allows the predictor to compute the tag scores on the basis of the user's and the image's representations. The convolutional block of the CNN  $f^e$  is capably adopted in a tag recommendation model in which it supplies visual features of an image for the predictor. Indeed, given a post  $p_{u,i}$ , the extractor  $f^e$  receives the tensor  $R_i$  as its input to generate the visual representation of the  $i$ -th image.

$$z_i = f^e(R_i; \theta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m \quad (5.5)$$

The output of  $f^e$  is a  $m$ -dimensional dense vector  $z \in \mathbb{R}^m$  which is aggregated with the user's and the tag's information to procedure the post-based or triple-based input for the tag predictor.

### 5.4.3 Content-aware Factorization Machines

In our approach, we aim at taking advantage of the FM that allows the encoding of any additional data to extend the input features and the strong capability of a CNN to learn visual contents of images. The model architecture is described by Figure 5.2. According

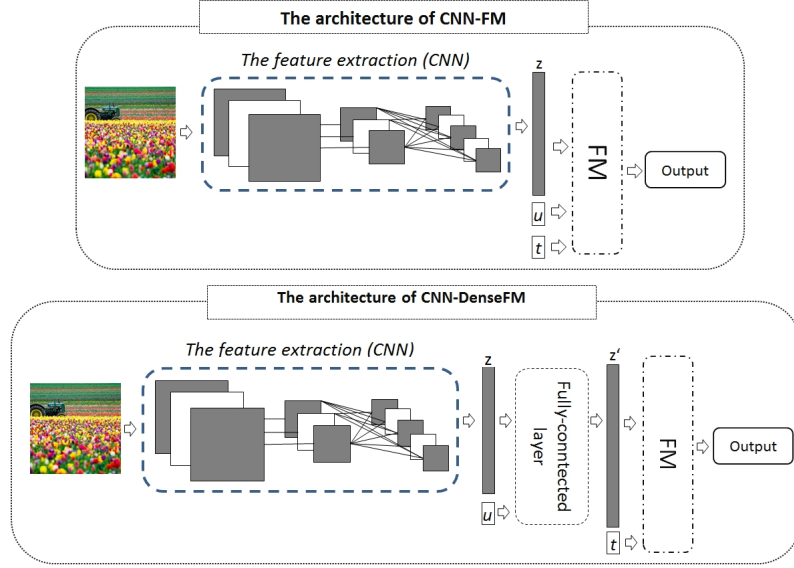


Figure 5.2: The architecture of CNN-FM and CNN-DenseFM

to the proposed architectures, a convolutional block, which consists of a series of three convolutional layers interspersed with two max pooling layers, is used as a visual feature generator  $f^e$  and an adaptive FM  $f^p$  following the block to compute scores of tags.

Candidate tags are recommended for a post  $(u, R_i)$  according to historical tagging information of the user  $u$  and visual contents of image  $i$ . For each pair  $(u, R_i)$ , a tensor  $R_i$  involving RGB values of the image  $i$  is passed through the CNN extractor to obtain a feature vector  $z_i \in \mathbb{R}^m$ . In addition, the user index,  $u$ , is converted to a one-hot vector defined as

$$\mathbf{x}_u = \phi(u) : \mathbb{N}^+ \rightarrow \{0, 1\}^{|U|} \quad (5.6)$$

Then, the representations of both the given user  $u$  and image  $i$ ,  $\mathbf{x}_u$  and  $z_i$ , are concatenated as an input, which is fed to a FM-based predictor to calculate all tags' scores directly or indirectly. The predictive process is summarized as

$$\hat{y}_{u,i} = f^p((\mathbf{x}_u, z_i); \omega) : \mathbb{R}^{|U|+m} \rightarrow \mathbb{R}^{|T|} \quad (5.7)$$

The output of the predictor is a score vector of tags in which each element is calculated based on the adapted FM function.

We examine the input description in Equation 5.3 as the combination of three sub-vectors associated with users, images and tags  $\mathbf{x}_{u,i,t} = (\mathbf{x}_u, \mathbf{x}_i, \mathbf{x}_t)$  where  $\mathbf{x}_u$ ,  $\mathbf{x}_i$  and  $\mathbf{x}_t$

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

are binary vectors encoding user's, image's and tag's indices. In addition, the interaction weights  $V$  in Equation 5.4 can be broken into three sub-variables  $V^U \in \mathbb{R}^{|U| \times p}$ ,  $V^I \in \mathbb{R}^{|I| \times p}$  and  $V^T \in \mathbb{R}^{|T| \times p}$  connected to users, images and tags.

Similarly, the weights of input features  $w$  are subdivided into  $w^U \in \mathbb{R}^{|U|}$ ,  $w^I \in \mathbb{R}^{|I|}$  and  $w^T \in \mathbb{R}^{|T|}$ . We use the  $'$  character to indicate the transpose operation. For example,  $x'_u, x'_i, x'_t$  are the transposes of  $x_u, x_i, x_t$ , and  $(V^U)', (V^I)', (V^T)'$  are the transposes of  $V^U, V^I, V^T$ . Equation 5.4 is rewritten as

$$(\hat{y}_{u,i})_t = w_0 + x'_u w^U + x'_i w^I + x'_t w^T + (x'_u V^U)((V^I)' x_i) + (x'_u V^U + x'_i V^I)((V^T)' x_t) \quad (5.8)$$

Because vectors  $x_u, x_i$ , and  $x_t$  are one-hot encoding vectors which have a single 1 value at  $u, i$  or  $t$  position and 0 at others, the product between these vectors and factor variables  $V$  are the  $u$ -th,  $i$ -th or  $t$ -th rows of  $V$ . Similarly, the product of these vectors and weight vector  $w$  are  $u$ -th or  $t$ -th elements of  $w$ . The one-hot encoding steps of users and tags are not necessary, and they are removed from the general process. The equation 5.8 can be shortened as

$$(\hat{y}_{u,i})_t = w_0 + w^U_u + w^I_i + w^T_t + \sum_{k=1}^p \left( V^U_{u,k} V^I_{i,k} + V^U_{u,k} V^T_{t,k} + V^T_{t,k} V^I_{i,k} \right) \quad (5.9)$$

It is applied to estimate each scoring element of the predicted output in our proposed approach through two possibly different methods.

1. **Direct way:** The model applied in this case is notated **CNN-FM**, which receives the aggregation of the visual image features, the user's and tag's one-hot representations as its input.

The visual vector  $z_i$  replaces the one-hot vector  $x_i$  in the input  $x$  of the predictor. The length of the weight vector  $w$  and the projection  $V$  are changed according to the size of visual features  $z$ , i.e.  $w^I \in \mathbb{R}^m$  and  $V^I \in \mathbb{R}^{m \times p}$ . The scoring function in Equation 5.9 becomes:

$$\begin{aligned} (\hat{y}_{u,i})_t = & w_0 + w^U_u + \sum_{a=1}^m (z_i)_a w^I_a + w^T_t \\ & + \sum_{k=1}^p \left( V^U_{u,k} \sum_{j=1}^m (z_i)_j V^I_{j,k} + V^U_{u,k} V^T_{t,k} + V^T_{t,k} \sum_{j=1}^m (z_i)_j V^I_{j,k} \right) + \varphi_i \end{aligned} \quad (5.10)$$

where  $\varphi_i$  is the interaction of visual features which does not appear in Equation 5.9. The reason is that the  $x_i$  has only one non-zero element, but  $z_i$ , a dense vector, contains many non-zero pairs of elements. The  $\varphi_i$  can be denoted as

$$\varphi_i = \sum_{a=1}^{m-1} \sum_{b=a+1}^m (z_i)_a (z_i)_b \sum_{k=1}^p V^I_{a,k} V^I_{b,k} \quad (5.11)$$

where  $V^I_{a,k}, V^I_{b,k}$  are elements at the position  $(a, k)$  and  $(b, k)$  of  $V^I$ .

2. **Indirect way:** The model applied in this case is called **CNN-DenseFM**, which obtains the representative features for both the user and the image before feeding them to the scoring function with the one-hot representation of the tag.

Instead of using the one-hot vector  $\mathbf{x}_u$  and the visual vector  $\mathbf{z}_i$ , we seek to learn latent features representing the combination of these features  $\mathbf{h}_{u,i}$ . To find the features, we find a transfer function

$$\mathbf{o}_{u,i} = f^{tr}((\mathbf{x}_u, \mathbf{z}_i); W, b) : \mathbb{R}^{|U|+m} \rightarrow \mathbb{R}^n \quad (5.12)$$

where  $n$  is the size of latent features;  $W$  and  $b$  parameterize the function. In fact, the function is considered as a fully-connected layer that has an half-sparse input  $\mathbf{h}_{u,i} = (\mathbf{x}_u, \mathbf{z}_i)$ . The layer can be defined as

$$\mathbf{o}_{u,i} = f^{tr}(\mathbf{h}_{u,i}) = \sigma(W\mathbf{h}_{u,i} + b) \quad (5.13)$$

where  $W \in \mathbb{R}^{n \times (|U|+m)}$  is a weight matrix and  $b \in \mathbb{R}^n$  is the bias of the input  $\mathbf{h}_{u,i}$ ;  $\sigma$  is an activation function. In the proposed approach, **ReLU** is used as the activation function. The weight vector  $\mathbf{w}^{UI} \in \mathbb{R}^n$  replaces two weight vectors of users  $\mathbf{w}^U$  and images  $\mathbf{w}^I$ . Similarly,  $V^{UI} \in \mathbb{R}^{n \times p}$  puts in place of  $V^U$  and  $V^I$ . Later, the scoring function is derived from Equation 5.9 as

$$(\hat{y}_{u,i})_t = w_0 + \sum_{j=1}^n (\mathbf{o}_{u,i})_j \mathbf{w}_j^{UI} + w_t^T + \sum_{k=1}^p V_{t,k}^T \sum_{j=1}^n (\mathbf{o}_{u,i})_j V_{j,k}^{UI} + \varphi_{ui} \quad (5.14)$$

where  $\varphi_{ui}$  is the interaction of latent features with respect to the representation of the post. It is defined as

$$\varphi_{ui} = \sum_{a=1}^{n-1} \sum_{b=a+1}^n (\mathbf{o}_{u,i})_a \cdot (\mathbf{o}_{u,i})_b \sum_{k=1}^p V_{a,k}^{UI} \cdot V_{b,k}^{UI} \quad (5.15)$$

#### 5.4.4 Optimization

The parameters  $\Theta$  of the recommendation model are learned by maximizing the Bayesian Personalized Ranking (BPR) optimization criterion [119]. According to this approach, the criterion with respect to the predicted scores of relevant and irrelevant tags is defined as:

$$\text{BPR-OPT}(\hat{\Theta}) := \sum_{(u,i,t^+,t^-) \in \mathcal{D}^{train}} \ln \sigma((\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-}) \quad (5.16)$$

The criterion is built with respect to the quadruple  $(u, i, t^+, t^-)$  where the tag  $t^+$  is assigned to the post  $(u, i)$  and the unobserved tag  $t^-$  is not used to annotate this post.

The proposed approach receives a post  $(u, i)$  as its input and the output is a  $|T|$ -dimensional vector including scores of all tags in the dataset. For this reason, we adapt the criterion with respect to posts and define it as

$$\text{BPR-OPT}(\hat{\Theta}) := \sum_{(u, R_i) \in \mathcal{P}^{train}} \frac{1}{|T_{u,i}^+| |T_{u,i}^-|} \sum_{t^+ \in T_{u,i}^+} \sum_{t^- \in T_{u,i}^-} \ln \sigma((\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-}) \quad (5.17)$$

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

The set of tags that the user  $u$  has assigned to the image  $i$  is denoted as  $T_{u,i}^+ := \{t \in T | (u, i, t) \in \mathcal{S}^{train}\}$  and the set of unobserved tags is defined as  $T_{u,i}^- := \{t \in T | (u, i, t) \notin \mathcal{S}^{train}\}$ .

When the pairwise loss is applied, the overlapping part in the predictive function is dismissed. As the result, Equation 5.10 can be re-formulated as

$$(\hat{y}_{u,i})_t = w_t^T + \sum_{k=1}^p \left( V_{u,k}^U + \sum_{j=1}^m (z_i)_j V_{j,k}^I \right) V_{t,k}^T \quad (5.18)$$

Similarly, Equation 5.14 can be re-written as

$$(\hat{y}_{u,i})_t = w_t^T + \sum_{k=1}^p V_{t,k}^T \sum_{j=1}^n (o_{u,i})_j V_{j,k}^{UI} \quad (5.19)$$

A stochastic gradient descent optimizer (SGD) is applied in the training process, where the gradient used for updating parameters is calculated per post  $(u, i)$  drawn randomly from the training set. The parameters of both the extractor  $\theta$  and predictor  $\omega$  are updated based on the derivative of BPR-OPT given a case  $(u, i)$  with respect to these parameters

$$\theta_{t+1} = \theta_t + \alpha_e \frac{\partial \text{BPR-OPT}}{\partial \theta_t} \quad (5.20)$$

$$\omega_{t+1} = \omega_t + \alpha_p \frac{\partial \text{BPR-OPT}}{\partial \omega_t} \quad (5.21)$$

where  $\alpha_e$  and  $\alpha_p$  are the learning rate values used to update the extraction and prediction parameters. The derivative of BPR-OPT for the given post with respect to a parameter  $\omega$  included in the predictor is

$$\frac{\partial \text{BPR-OPT}}{\partial \omega} = \frac{1}{|T_{u,i}^+| |T_{u,i}^-|} \sum_{t^+} \sum_{t^-} (1 - \sigma(\hat{y}_{t^+, t^-})) \left( \frac{\partial (\hat{y}_{u,i})_{t^+}}{\partial \omega} - \frac{\partial (\hat{y}_{u,i})_{t^-}}{\partial \omega} \right) \quad (5.22)$$

where  $\hat{y}_{t^+, t^-} = (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-}$ .

The derivative of the loss with respect to a parameter being in a specific layer depends on the error which is transferred backward from the next layer. For example, a parameter  $\theta^l$  of the last convolutional layer will receive the error term defined as

$$\begin{aligned} \delta_i &= \frac{\partial \text{BPR-OPT}}{\partial z_i} = \frac{1}{\underbrace{|T_{u,i}^+| |T_{u,i}^-|}_{\Gamma_{u,i}}} \sum_{t^+} \sum_{t^-} \underbrace{(1 - \sigma(\hat{y}_{t^+, t^-}))}_{\Psi_{u,i,t^+,t^-}} \left( \frac{\partial (\hat{y}_{u,i})_{t^+}}{\partial z_i} - \frac{\partial (\hat{y}_{u,i})_{t^-}}{\partial z_i} \right) \\ &= \Gamma_{u,i} \sum_{t^+} \sum_{t^-} \Psi_{u,i,t^+,t^-} V^I (V_{t^+}^T - V_{t^-}^T) \end{aligned} \quad (5.23)$$

Then, the derivative of  $\theta^l$  will be presented as

$$\frac{\partial \text{BPR-OPT}}{\partial \theta^l} = \delta_i \frac{\partial z_i}{\partial \theta^l}$$

In the case of **CNN-DenseFM**, visual features  $z_i$  are transferred through a fully-connected layer before going to the FM-based predictor. For this reason, the error which the last convolutional layer gets from the predictor can be seen as

$$\begin{aligned}\delta_i &= \Gamma_{u,i} \sum_{t^+} \sum_{t^-} \Psi_{u,i,t^+,t^-} \left( \frac{\partial(\hat{y}_{u,i})_{t^+}}{\partial o_{u,i}} - \frac{\partial(\hat{y}_{u,i})_{t^-}}{\partial o_{u,i}} \right) \frac{\partial o_{u,i}}{\partial z_i} \\ &= \Gamma_{u,i} \sum_{t^+} \sum_{t^-} \Psi_{u,i,t^+,t^-} V^{UI}(V_{t^+}^T - V_{t^-}^T) \frac{\partial o_{u,i}}{\partial z_i}\end{aligned}$$

The learning process is described in Algorithm 1. For each crawled post, the model predicts the tag scores for the post. The gradients of the loss with respect to the model parameters are found and used to update the parameters.

---

**Algorithm 1** Learning BPR

---

- 1: **Input:**  $\mathcal{P}^{train}, \mathcal{S}^{train}, \alpha_e, \alpha_p$
  - 2: **Output:**  $\theta, \omega$
  - 3:  $\theta \leftarrow \mathcal{N}(0, 0.1)$
  - 4:  $\omega \leftarrow \mathcal{N}(0, 0.1)$
  - 5: **repeat**
  - 6:   Pick  $(u, R_i) \in \mathcal{P}^{train}$  randomly
  - 7:   Get  $T_{u,i}^+$  and  $T_{u,i}^-$
  - 8:    $z_i \leftarrow f^e(R_i)$
  - 9:   **for**  $t \in 1, \dots, |T|$  **do**
  - 10:     Compute  $(\hat{y}_{u,i})_t$
  - 11:   **end for**
  - 12:   Compute BPR $_{u,i}$  according to Equation 5.17
  - 13:    $\Delta_\omega \leftarrow \frac{\partial \text{BPR}}{\partial \omega}$
  - 14:    $\Delta_\theta \leftarrow \frac{\partial \text{BPR}}{\partial \theta}$
  - 15:   Update  $\theta \leftarrow \theta + \alpha_e \Delta_\theta$
  - 16:   Update  $\omega \leftarrow \omega + \alpha_p \Delta_\omega$
  - 17: **until** convergence
  - 18: **return**  $\theta, \omega$
- 

## 5.5 Evaluation

We performed experiments addressing the impact of visual contents on the recommendation process. In Section 5.5.2, we describe the CNN architecture adopted for feature extraction, the evaluation methodology and settings of the proposed models. The results of the evaluation are detailed in Section 5.5.3.

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

---

### 5.5.1 Dataset

We conducted experiments on **LOPO-2** and **LOPO-5**, which were implemented in Chapter 4. Since LOPO-2 contains several rare users, which occur once during the learning process, LOPO-5 are used to evaluate the proposed models so that each user in the test set has been learned through at least four instances.

### 5.5.2 Experimental Setup

The CNN architecture used in the experiments contains 3 convolutional layers. The first layer filters a  $75 \times 75 \times 3$  input image with 10 kernels of size  $6 \times 6 \times 3$  with a stride of 3 pixels. The input of the second layer is the output of the pooling layer that takes the maximum value of a square block  $2 \times 2$  from the first convolutional layer to generate one value of each feature map. It uses 30 kernels of size  $6 \times 6 \times 10$  with a stride of 2 pixels to produce feature maps for the next connected pooling layer. The following pooling layer also uses max pooling to produce values of feature maps. The last convolutional layer filters 30 feature maps with 128 kernels of size  $2 \times 2 \times 30$  to provide a feature vector for the predictor. In the CNN-DenseFM, a fully-connected layer that extracts the combination features of a post  $(u, i)$  produces a dense vector having 128 elements for the recommender.

Due to scalability issues, we evaluated the CNN-DenseFM model only on the LOPO-5 protocol. On the other hand, the CNN-FM model was evaluated in both LOPO-2 and LOPO-5 protocols. For the evaluation, we used three metrics to capture the performance of the models as proposed by Rendle & Schmidt-Thieme [117].

- Precision at rank K

$$\text{Precision@K} = \text{avg}_{(u,i) \in S_{test}} \frac{|\hat{T}_{u,i}^K \cap T_{u,i}^+|}{K}$$

- Recall at rank K

$$\text{Recall@K} = \text{avg}_{(u,i) \in S_{test}} \frac{|\hat{T}_{u,i}^K \cap T_{u,i}^+|}{|T_{u,i}^+|}$$

- F1-measure at rank K

$$\text{F1@K} = \frac{2 \cdot \text{Precision@K} \cdot \text{Recall@K}}{\text{Precision@K} + \text{Recall@K}} \quad (5.24)$$

The number and the size of CNN kernels are fixed like above. We applied a grid search mechanism for looking for the best learning rate  $\alpha$ , regularization  $\lambda$  and factor dimension  $p$ . The learning rate is selected among the range of  $\alpha_{cnn/full} \in \{1e-2, 1e-3, 1e-4\}$  for all convolutional layers and the fully-connected layer,  $\alpha_{fm} \in \{1e-2, 1e-3, 1e-4\}$  for the FM layer. The values of the regularization hyperparameter for all convolutional layers, the fully-connected layer and the FM layer are selected from  $\lambda_{cnn/full} \in \{1e-03, 1e-$



$04, 1e - 05\}$  and  $\lambda_{fm} \in \{1e - 05, 1e - 06, 1e - 07\}$ . The factor dimensions are searched among the range of  $p \in \{64, 128, 256\}$ .

We compare the proposed models with the following tag recommendation models: **PITF** [117], **FM** [115], Most popular tags (**MP**) [56], Adapted PageRank (**AP**) [55], FolkRank (**FR**) [56].

We used the Tagrec framework built by Kowald *et al.* [67] to learn MP, FR and AP while the Tag recommender software<sup>1</sup> was used to learn PITF and FM. We also implemented a convolutional neural network with a multilayer perceptron network placed as the last layer (**CNN-MLP**) to recommend tags based on the content of images. The convolutional block having the same architecture used in the CNN-FM is deployed in the first step to extract a feature vector, then its values are fed to a multilayer perceptron network as the input to calculate scores of all tags. The parameters of this model are learned based on the pairwise loss function.

### 5.5.3 Results

The MP recommends the top tags having the highest usage frequencies according to all images and ignores the user’s preferences and the content of resources be tagged. It does not recommend specific tags for a particular user or image so many tags are irrelevant to a given post. For this reason, the performance of the MP is the worst compared to other approaches, as shown in Figure 5.3 and Figure 5.4.

Although the content-based tag recommendation (CNN-MLP), which predicts tags associated with the visual information of images, works better than the MP, its recommended tags are not specific for users, i.e the tag sets suggested for similar images are the same. However, in practice, different users assign distinct keywords to related images depending on their personality. For this reason, its performance cannot compete with other personalized models. This means that the content-based tag recommendation is not really effective in the case of personalized tag recommendation.

On the contrary, visual features strongly support the personalized models to achieve higher performance indicated by Figure 5.3, Figure 5.4 and Table 5.1. For example, the recommender using visual contents improves the performance by 7.2 percent of precision at 1 in LOPO-2 and 3.8 percent in LOPO-5 if compared to FM. Although Rendle [115] discussed that the performance of the FM and the PITF are comparable in ECML Discovery Challenge 2009 (task 2), the PITF achieves higher performance than the FM does in this narrow folksonomy when the latent features of tags are specified for different interactions between them and users or images. Feeding a FM with CNN-extracted visual features boosts its performance, and it is comparable to a PITF. The result also opens the direction of using visual contents with the PITF model.

The proposed approach relies on the interaction of visual features and tags as well as users and tags; additionally, any image is able to be represented by a visual feature vector. For a new image, the model is still capable to capture the interaction between it and tags

<sup>1</sup><http://www.informatik.uni-konstanz.de/rendle/software/tag-recommender/>

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

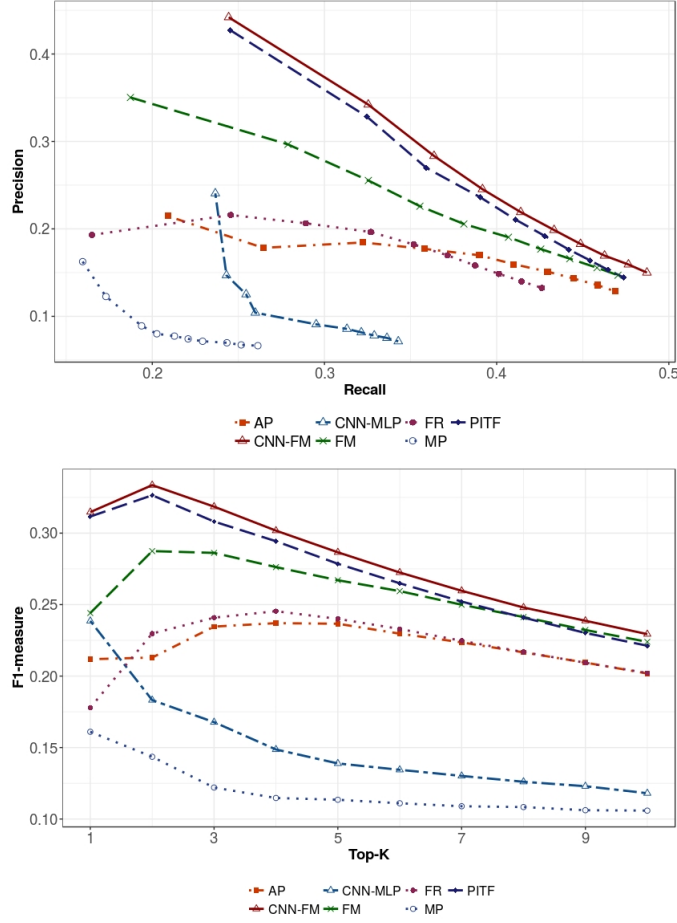


Figure 5.3: Precision, Recall and F1-Measure for LOPO-2 from rank 1 to 10

besides the relationship of a given user and tags while the FM or PITF mostly counts on the user-tag relation to predict tags. From this point of view, the proposed approach can be considered to solve the cold-start image recommendation problem.

Depending on the feeding types, the direct or indirect way, the combination of users and visual features can improve the prediction capability of the recommender in different level. It is clearly visible through the performance of CNN-DenseFM in Figure 5.4. When the intermediate step is used to combine user's and visual features into a general representation, the performance of the recommender is worse than the recommender receiving these features directly. One reason for this problem can be that the visual values overwhelm the one-hot user values during the combining process. The generated features are more related to the contents of images and loose user information.

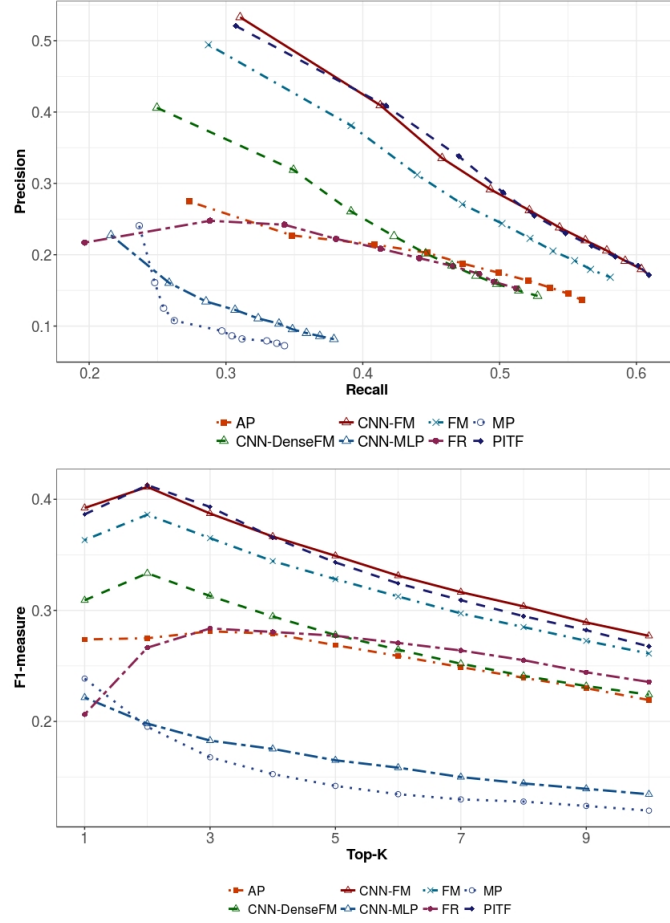


Figure 5.4: Precision, Recall and F1-Measure for LOPO-5 from rank 1 to 10

## 5.6 Discussion

We proposed a hybrid model, which depends on the CNN and FM architectures, to calculate scores of tags. Empirically, the learnable features of images strengthen the performance of the tag recommendation, which captures the interaction among users, image’s features, and tags. The representation of a given post, which is generated by aggregating a one-hot and a visual representation, does not boost up the performance of the model. However, it raises a question whether the interaction between each feature of a user and an image used as the post representation is useful for the tag recommendation. Moreover, instead of using the combined model to predict tags, a full CNN architecture, in which a multi-layer perceptron replaces the FM’s mission, is a promising method for the personalized tag recommendation. Additionally, the experiments show that PITF is a next candidate to combine with CNN-based models so that the performance will be expected

## 5. FACTORIZATION CONTENT-AWARE TAG RECOMMENDATION

---

Table 5.1: F1-measure at 5 and 10

Models	LOPO-2 (F1@5)	LOPO-2 (F1@10)	LOPO-5 (F1@5)	LOPO-5 (F1@10)
MP	0.1135	0.1059	0.142	0.1197
CNN-MLP	0.1389	0.1181	0.165	0.1344
AP	0.2366	0.2018	0.2686	0.2193
FR	0.24	0.2021	0.2772	0.2356
CNN-DenseFM	(N/A)	(N/A)	0.2778	0.2241
FM	0.2671	0.224	0.3281	0.2611
PITF	0.2786	0.2242	0.3432	0.2675
CNN-FM	0.2866	0.2293	0.3491	0.2771

better than history-based models.

The learning time is also an issue for the model when most time is spent on backpropagation and updating the convolutional block. Furthermore, the training data size also limits the depth of the CNN. A supporting technique should be aggregated with the tag recommendation so that it allows a very deep model to be applied as the extractor, and the time will be concentrated for learning the predictor.

## Chapter 6

# Personalized Deep Learning for Tag Recommendation

### Contents

---

<b>6.1</b>	<b>Introduction . . . . .</b>	<b>49</b>
<b>6.2</b>	<b>Problem Extension . . . . .</b>	<b>51</b>
<b>6.3</b>	<b>Personalized Content-aware Tag Recommendation . . . . .</b>	<b>52</b>
6.3.1	Convolutional Neural Network as the Extractor . . . . .	52
6.3.2	Personalized Fully-connected Layer . . . . .	53
6.3.3	Multilayer Perceptron as the Predictor . . . . .	54
6.3.4	Personalized Convolutional Layer . . . . .	54
6.3.5	Optimization . . . . .	55
<b>6.4</b>	<b>Evaluation . . . . .</b>	<b>56</b>
6.4.1	Dataset . . . . .	57
6.4.2	Experimental Setup . . . . .	57
6.4.3	Results . . . . .	58
<b>6.5</b>	<b>Discussion . . . . .</b>	<b>62</b>

---

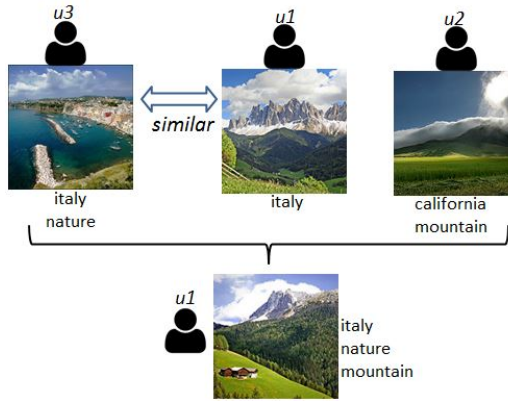
### 6.1 Introduction

In the previous chapter, we have shown that good results can be achieved by employing visual representation of images in a tag recommendation model. A FM is adapted to compute scores of tags based on capturing the interaction among users, visual features of images and tags. A plainly personalized model, such as PITF, has difficulties to predict tags for new images because of the missing of the relation between them and other entities. Mostly, it suggests the most popular tags of a given user to an image. On the

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

contrary, the earlier proposed approach (CNN-FM) seeks to learn the relation between visual features and others, and any image can be represented by a feature vector, even new images that have not appeared in the training data. Recommended tags of a personalized content-aware tag recommendation, like CNN-FM, express personal and content-aware characteristics as in Figure 6.1. Tags recommended for  $u1$  contain his favorite word *italy*, a word *mountain* related to the content of the image and a word *nature* from  $u3$  being similar to  $u1$ .

Figure 6.1: Characteristics of tags predicted by a personalized content-aware approach



This approach can be seen as a hybrid model taking advantage of both the state-of-art approaches in tag recommendation and image classification. Compared to CNN-based architectures which are used to solve the multi-label classification or content-based tag recommendation [41, 109, 147, 149], the model also have a convolutional block to extract visual features of images to be used in the predictor. While the model proposed by Wei *et al.* [149] aggregates the scores of labels from different segment hypotheses by using the max pooling, Gong *et al.* [41] proposed to learn the CNN model for the multi-label classification with respect to different losses including the pairwise loss and a ranking loss, WARP [151]. The combination of a deep CNN and a recurrent neural network is used by Wang *et al.* [147] to learn the dependencies of labels in an image. Recently, Rawat & Kankanhalli [109] combine a deep CNN with a neural network to enrich the visual image features with the context information extracted from the meta-data including time and geographical data.

These approaches are based on the image's information to predict probabilities of labels so recommended tags only reflect the content of images and do not have any connection with the user's preferences. Differently, the prediction layer of the proposed approach allows user information going through such that the interaction among users, image representation, and tags is captured. If a CNN model is capable to obtain personalized visual features, it can predict a set of tags related to both the user's preferences and the image's contents.

In this chapter, we show how a deep learning approach can be adapted to solve a

personalized image tag recommendation. For a personalized problem, features used in a deep learning model have to include the information of a user and an associated image. We propose a new way to add the user's information into the CNN models. A new layer that captures the interaction between users and visual features plays a bridging role between a CNN image extractor and a multilayer perceptron predictor as in Figure 6.2.

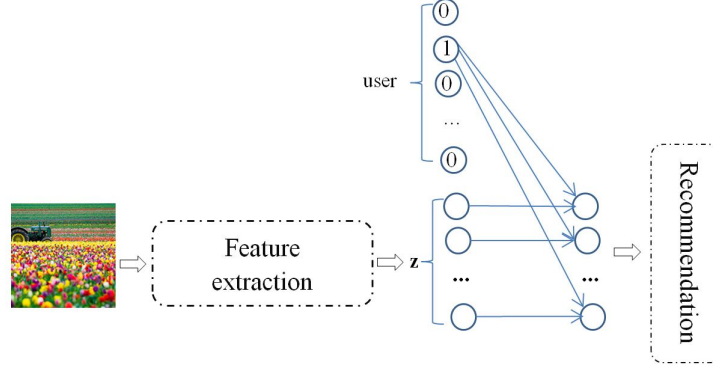


Figure 6.2: The architecture of CNN-PerMLP

In addition, we adopt the BPR in a different way to put the model into use. Empirically, our experiments were conducted in two real datasets, namely NUS-WIDE and Flickr-PTR, show that the proposed model outperforms the state-of-the-art personalized tag recommendation models, which are purely based on tagging history, up to at least four percent. The experiments also indicate the stronger support of the supervised features to increase the prediction quality up to at least two percent compared to low-level features.

## 6.2 Problem Extension

As the previous formulation for the tag recommendation problem, we are exploiting the relation between users, images represented by visual features and tags.

To increase instances of the dataset, we crop images at different positions to generate several patches and denote  $R_i^q$  being a patch  $q$  of an image  $i$ . So, a post  $(u, i)$  has  $Q$  variants based on patches of images, and the total number of training instances will be  $Q \times |\mathcal{P}^{train}|$ . The color image set is re-defined as

$$\mathcal{R} = \{R_i^q | i \in I \wedge R_i^q \in \mathbb{R}^{d \times d \times 3}\} \quad (6.1)$$

Each post is annotated with a set of tags, known as relevant tags,  $T_{u,i}^+ \in T$  and the rest of unobserved tags are grouped in the irrelevant tag set  $T_{u,i}^- \in T \setminus T_{u,i}^+$ .

We also seek to learn a machine learning model

$$\hat{y}_{u,i,q} = f(u, R_i^q; \Theta) : U \times \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^{|T|} \quad (6.2)$$

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

which maps an instance of a given post to real-valued scores of tags. Scores of a post are the average scores of its instances according to the patches of the used image

$$\hat{y}_{u,i} = \frac{1}{Q} \sum_{q=1}^Q \hat{y}_{u,i,q} \quad (6.3)$$

### 6.3 Personalized Content-aware Tag Recommendation

The architecture of the proposed model, called **CNN-PerMLP**, based on the relation between the user and the visual features of the given image is illustrated in Figure 6.2. The supervised visual features are computed by passing a patch  $q$  of the image  $i$  through the CNN feature extractor. To personalize the visual features, a proposed specific layer called the personalized fully-connected layer is employed following the extractor. The layer captures the interaction between the user and each visual feature to generate the latent features for the post  $p_{u,i}$ . A neural network is deployed as a predictor to compute the relevant probabilities of tags. The network receives the user-image features as the input and its outputs are used to derive a ranking of recommended tags.

According to the architecture, we can describe the model function  $f$  by a sequence of operations

$$\hat{y}_{u,i,q} = f(u, R_i^q; \Theta) = f^p \left( f^u(u, f^e(R_i^q; \theta); \vartheta); \omega \right) \quad (6.4)$$

where  $f^e$  is the extractor which provides visual features of an image  $R_i^q$ ,  $f^u$  is responsible to capture the interaction between users and visual features and  $f^p$  is used to compute scores of tags for the given post. Moreover,  $\theta$ ,  $\vartheta$ , and  $\omega$  are parameters of the extractor, the personalized layer, and the predictor respectively.

Besides, several alternative forms of the proposed model, which personalize feature maps at the different levels instead of adding the user's information into the last representation of an image, are exploited to demonstrate the effectiveness of the personalization level. The processes can be formulated as

$$\hat{y}_{u,i,q} = f(u, R_i^q; \Theta) = f^p \left( f^{ue}(u, R_i^q; \theta, \vartheta); \omega \right) \quad (6.5)$$

where  $f^{ue}$  is the personalized extractor which outputs the representation of a post  $(u, R_i^q)$ . In the test phase, we average scores of all instances correlated with the given post as mentioned in Equation 6.3.

#### 6.3.1 Convolutional Neural Network as the Extractor

Proving the powerful capability to extract visual features of images through a variety of studies [50, 68, 131] and the previous section results, we continue using a CNN as the feature extractor. For the later reference, we re-formulate the extracting process based on



### 6.3 Personalized Content-aware Tag Recommendation

the convolutional network. Generally, more than one convolutional layers are employed to generate several feature maps by moving kernel windows smoothly across images.

For example, the  $k$ -th feature map of a given layer is denoted as  $\tau^k$ , the weights and the biases of the filters for  $\tau^k$  are  $W^k \in \mathbb{R}^{p_1 \times \mathbb{R}^{p_2} \times \mathbb{R}^{p_2}}$  and  $b_k$  where  $p_1$  is the number of the previous layer's feature maps and  $p_2$  is the dimension of kernel windows. The element at the position  $(i, j)$  of  $\tau^k$  is acquired as

$$\tau_{ij}^k = \sigma \left( b_k + \sum_{a=1}^{p_1} (W_a^k * \xi^a)_{ij} \right) \quad (6.6)$$

with  $*$  being the convolution operator,  $\xi^a$  being the  $a$ -th feature map of the previous layer and  $\sigma$  being the activation function. The pooling layer, which maximizes or averages a rectangular block of the previous layer to provide an element for the current feature map, follows the convolutional layer. If the max pooling operator is used, the element at the position  $(i, j)$  of the  $k$ -th feature map  $\tau^k$  is denoted as

$$\tau_{ij}^k = \max_{a,b} (\xi^k)_{a,b} \quad (6.7)$$

where  $a$  and  $b$  are the positions of the element associating to the pooled block. The output of the CNN feature extractor is a dense feature vector representing the visual content of the image. We define the extraction process of the patch  $q$  of the  $i$ -th image by

$$z_{i,q} = f^e(R_i^q; \theta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m \quad (6.8)$$

where  $\theta$  is the parameter set of the extractor. These features only contain the visual values of the given image and there is no user's information included in the representation. To personalize visual features, a layer should be employed immediately after the extractor.

#### 6.3.2 Personalized Fully-connected Layer

As we mentioned in the above section, the extracted features from the convolutional block only contain the information of a given image  $i$ . To make visual features specific for a given user, the user's information has to be added or combined with them. For this reason, a layer between the feature extractor and the predictor is utilized to generate the user-aware features that are used as the input of the predictor.

In the previous chapter, the concatenation between the visual and one-hot features does not create a useful representation for the predictor. In this proposed approach, the user's index is used to obtain latent features paired with the image's ones. Given a  $u$ -th user described as a one-hot vector encoding the user's id and being represented as

$$x_u = \phi(u) : \mathbb{N}^+ \rightarrow \{0, 1\}^{|U|}$$

both the visual feature vector  $z_i^q$  and the sparse vector  $x_u$  are the inputs of this layer. The layer seeks to learn a function to map each pair of users and image features to a personalized feature vector and it can be denoted as

$$h_{u,i,q} = f^u(z_{i,q}, x_u; \vartheta) : \mathbb{R}^m \times \{0, 1\}^{|U|} \rightarrow \mathbb{R}^m \quad (6.9)$$

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

In fact, it will capture the association between the user and each visual feature, and it is obtained as follows

$$h_{u,i,q} = f^u(z_{i,q}, \mathbf{x}_u; \vartheta) = \sigma(b^u + \mathbf{w}^u \odot z_{i,q} + V\mathbf{x}_u) \quad (6.10)$$

where  $\mathbf{w}^u \in \mathbb{R}^m$  is the weights of the visual features,  $V \in \mathbb{R}^{m \times |U|}$  is the weights of the user features and  $\odot$  is Hadamard product. As in the convolutional layer, the elementwise activation function  $\sigma$  is applied to the linear combination of the weighted visual feature and the user's features. The output elements representing both the user's and image's characteristics are ready to be transferred to the predictor in which scores of tags are estimated.

### 6.3.3 Multilayer Perceptron as the Predictor

To compute scores of tags, a multilayer perceptron with one hidden layer is adopted as the predictor and its input is the output of the personalized fully-connected layer  $h_{u,i,q}$ . The output of the network is a  $|T|$ -dimensional vector containing score values of tags associated with the post  $(u, i)$  and the patch  $q$  of the image  $i$ . In summary, the model maps a post  $(u, i)$  represented by a personalized feature vector  $h_{u,i,q}$  to real-valued scores of tags and it is defined as

$$\hat{y}_{u,i,q} = f^p(h_{u,i,q}; \omega) : \mathbb{R}^m \rightarrow \mathbb{R}^{|T|} \quad (6.11)$$

Similar to other neural networks, the prediction function can be expanded as the chain of non-linear functions. For simplicity,  $h$  replacing  $h_{u,i,q}$  will be used in the following denotation of the predictor and we define the scoring function of the predictor as

$$\hat{y}_{u,i,q} = f^p(h_{u,i,q}; \omega) = \sigma\left(W^{out} \cdot \sigma(W^{hid}h^T + b^{hid})^T + b^{out}\right) \quad (6.12)$$

where  $W^{hid} \in \mathbb{R}^{n \times m}$  and  $b^{hid} \in \mathbb{R}^n$  are weights and biases of the hidden layer having  $n$  neurons;  $W^{out} \in \mathbb{R}^{|T| \times n}$  and  $b^{out} \in \mathbb{R}^{|T|}$  are weights and biases of the output layer.  $\sigma$  is the activation function which is often used in most neural networks.

### 6.3.4 Personalized Convolutional Layer

The aggregating protocol relying on the personalized fully-connected layer can be applied to personalize feature maps which are the output elements of a convolutional layer. Instead of using the personal layer after the last convolution layer, the architecture can be modified by moving this layer backward. By doing in this way, the personalized fully connected layer is adjusted so that it can take a 4-dimensional tensor  $\xi$  as an input and it is named the personalized convolutional layer.

It will personalize all feature maps by adding the user information to each pixel of feature maps  $\xi^l$  in the layer  $l$ . The process can be presented as

$$\tau_{u,i}^l = f^u(\mathbf{x}_u, \xi^l; \vartheta) : \mathbb{R}^{h \times w \times c} \times \{0, 1\}^{|U|} \rightarrow \mathbb{R}^{h \times w \times c}$$

## 6.3 Personalized Content-aware Tag Recommendation

where  $h, w$  are the height and width of a feature map, and  $c$  is the number of feature maps. Normally, feature maps are squared so  $h = w$ . In addition,  $\xi \in \mathbb{R}^{h \times w \times c}$  consists of feature maps generated by the previous convolution layer. The weights of the given user and pixels are shared between elements of the same feature map. In detail, if we denote the output of this layer as  $\tau$ , it can be computed as

$$\tau_{u,i}^l = f^u(u, \xi^l; \vartheta) = \sigma(b^u + w^u \odot \xi^l + Vx_u) \quad (6.13)$$

where  $w^u \in \mathbb{R}^c$  is the weights of feature maps and  $V \in \mathbb{R}^{c \times |U|}$  is the weights of users. The kind of layers can follow several convolutional layers to guarantee that the user information will not be lost in the feedforward route.

### 6.3.5 Optimization

We adopt the BPR optimization criterion in a different way so that the algorithm can be applied to learn the deep learning personalized image tag recommendation. In the

---

#### Algorithm 2 Learning BPR

---

```

1: Input:  $\mathcal{P}_{S_{train}}, \mathcal{S}_{train}, \mathcal{R}, N, \alpha$ 
2: Output:  $\Theta$ 

3: Initialize  $\Theta \leftarrow \mathcal{N}(0, 0.1)$ 
4: repeat
5:   Pick  $(u, i) \in \mathcal{P}_{S_{train}}$  and  $R_i^q \in \mathcal{R}$  randomly
6:   Get  $T_{u,i}^+ := \{t \in T \mid (u, i, t) \in \mathcal{S}_{train}\}$ 
7:   Sample  $\tilde{T}_{u,i}^- := \{t \in T \mid (u, i, t) \notin \mathcal{S}_{train}\}$  where  $|\tilde{T}_{u,i}^-| = N$ 
8:    $z_{i,q} \leftarrow f^e(R_i^q)$ 
9:    $h_{u,i,q} \leftarrow f^u(z_{i,q}, x_u)$ 
10:   $\hat{y}_{u,i,q} \leftarrow f^p(h_{u,i,q})$ 
11:  Initialize  $\text{BPR}(\Theta)_{u,i,q} = 0$ 
12:  for  $t^+ \in T_{u,i}^+$  do
13:    for  $t^- \in \tilde{T}_{u,i}^-$  do
14:       $\text{BPR}(\Theta)_{u,i,q} \leftarrow \text{BPR}(\Theta)_{u,i,q} + \ln \sigma((\hat{y}_{u,i,q})_{t^+} - (\hat{y}_{u,i,q})_{t^-})$ 
15:    end for
16:  end for
17:   $\text{BPR}(\Theta)_{u,i,q} \leftarrow \frac{\text{BPR}(\Theta)_{u,i,q}}{|T_{u,i}^+| \cdot |\tilde{T}_{u,i}^-|}$ 
18:  Backpropagate the error  $\delta^*$  of all layers w.r.t  $\text{BPR}(\Theta)_{u,i,q}$ 
19:   $\Delta_{\Theta} \leftarrow \frac{\partial \text{BPR}(\Theta)_{u,i,q}}{\partial \Theta}$ 
20:  Update  $\Theta \leftarrow \Theta + \alpha \Delta_{\Theta}$ 
21: until convergence
22: return  $\Theta$ 

```

---

proposed approach, the backpropagation algorithm is applied in which the errors will be

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

transferred backward to the input. If the original BPR is applied, the errors relying on the difference of two tags become smaller through several layers and the gradients of the loss with respect to convolution parameters are too small to update. The optimization of the criterion is not sufficient if applied directly to the model. The BPR criterion with respect to posts is proposed to use, and in terms of an instance  $q$  of a given post  $(u, i)$ , it is defined as in Equation 5.17.

Motivated by Dropout techniques where several nodes are dropped from the network, we remove a subset of irrelevant outputs instead of dropping out hidden units. In fact, we sample  $N$  tags from the unobserved set  $T_{u,i}^-$ , rather than using the entire rest, and only compute the loss based on these sampled tags. For simplicity,  $\tilde{T}_{u,i}^-$  is a denotation for the sampled irrelevant tags and  $|\tilde{T}_{u,i}^-| = N$ . The learning process is described in Algorithm 2. For each random post, an arbitrary patch of the associated image is chosen to extract the visual features. An irrelevant set having  $N$  tags is selected at random from the unobserved tags of the post. The system computes the scores of all relevant tags and the drawn irrelevant tags. From Equation 5.17, the gradient of BPR with respect to the model parameters is obtained as follows:

$$\frac{\partial \text{BPR}}{\partial \Theta} = \Omega \sum_{t^+ \in T_{u,i}^+} \sum_{t^- \in \tilde{T}_{u,i}^-} \Gamma_{t^+, t^-} \left( \frac{\partial (\hat{y}_{u,i,q})_{t^+}}{\partial \Theta} - \frac{\partial (\hat{y}_{u,i,q})_{t^-}}{\partial \Theta} \right) \quad (6.14)$$

where

$$\Omega = \frac{1}{|T_{u,i}^+| \|\tilde{T}_{u,i}^-\|} \quad \Gamma_{t^+, t^-} = 1 - \sigma((\hat{y}_{u,i,q})_{t^+} - (\hat{y}_{u,i,q})_{t^-}) \quad (6.15)$$

To learn the model, the gradients  $\frac{\partial (\hat{y}_{u,i,q})_{t^+}}{\partial \Theta}$  and  $\frac{\partial (\hat{y}_{u,i,q})_{t^-}}{\partial \Theta}$  have to be computed. Depending on the weights in the different layers, one or both the gradients are computed. For example, if the parameter  $\theta^l$  depends on the relevant tags  $t^+$ , Equation (6.14) becomes

$$\frac{\partial \text{BPR}}{\partial \theta^l} = \frac{\partial \text{BPR}}{\partial (\hat{y}_{u,i,q})_{t^+}} \times \frac{\partial (\hat{y}_{u,i,q})_{t^+}}{\partial \theta^l} = \frac{\partial (\hat{y}_{u,i,q})_{t^+}}{\partial \theta^l} \cdot \Omega \sum_{t^- \in \tilde{T}_{u,i}^-} \Gamma_{t_j^+, t^-} \quad (6.16)$$

To find the gradients of the CNN parameters, the derivatives with respect to the visual features are propagated backward to the CNN block. From the Equation 6.10 and 6.14, if we assume that the activation layer used in the layer is a linear function, the derivatives are defined by

$$\frac{\partial \text{BPR}}{\partial z_{i,q}} = \Omega \sum_{t^+ \in T_{u,i}^+} \sum_{t^- \in \tilde{T}_{u,i}^-} \Gamma_{t^+, t^-} \left( \frac{\partial (\hat{y}_{u,i,q})_{t^+}}{\partial h_{u,i,q}} - \frac{\partial (\hat{y}_{u,i,q})_{t^-}}{\partial h_{u,i,q}} \right) \odot w^{per} \quad (6.17)$$

### 6.4 Evaluation

In the evaluation, we performed experiments addressing the impact of supervised visual features and the personal factor on the tag recommendation process.

Table 6.1: The set of 10 most frequent tags in NUS-WIDE-1

blue	2.956	sky	2.810
water	2.667	clouds	2.255
green	2.227	red	2.137
sunset	1.818	nature	1.792
light	1.785	landscape	1.499

Table 6.2: The set of 10 most frequent tags in Flickr-PTR

white	2.139	blue	2.122
red	1.900	green	1.794
light	1.745	sky	1.583
water	1.578	bw	1.553
black	1.489	art	1.472

### 6.4.1 Dataset

We conducted the experiments on the NUS-WIDE-1 and Flickr-PTR dataset described in Chapter 4. For each digital image, four patches were generated as mentioned in Section 4.3. With the above filtering and splitting, the test set of NUS-WIDE contains around 99 percent unobserved images and in Flickr-PTR, images in the test set are completely new. Moreover, in these datasets, the color tags have high frequencies compared to other tags as shown in Table 6.1 and 6.2.

### 6.4.2 Experimental Setup

Table 6.3: Layer characteristics of the convolutional architectures

Layer	NUS-WIDE-1	Flickr-PTR
The first convolutional layer	$6 \times 6 \times 3$ (stride: 3)	$5 \times 5 \times 3$ (stride: 2)
The first max pooling layer	$2 \times 2$	$2 \times 2$
The second convolutional layer	$6 \times 6 \times 10$ (stride: 2)	$5 \times 5 \times 10$ (stride: 1)
The second max pooling layer	$2 \times 2$	$3 \times 3$
The third convolutional layer	$2 \times 2 \times 30$ (stride: 1)	$3 \times 3 \times 30$ (stride: 1)

The architectures used for both datasets contain 3 convolutional layers alternated with 2 max-pooling layers. The convolutional layers, *ConvLs*, in these architectures have the same number of kernels that are 10 for the first, 30 for the second and 128 for the third. Because of the difference of the image size, the dimensions of convolutional kernels and pooling blocks in these architectures are divergently shown in Table 6.3. The hidden layers of the predictor have the dimension 128 for both architectures and the ReLU function is used as the activation function.

The evaluation metric used in this chapter is F1-measure in top K tag lists as described in Equation 5.24. The grid search mechanism was used to find the best learning rate  $\alpha$  among the range  $\{1e-3, 1e-4, 1e-5\}$  for all *ConvLs* and  $\{1e-2, 1e-3, 1e-4\}$  for all fully-connected layers, the best L2-regularization  $\lambda$  from the range  $\lambda \in \{0.0, 1e-4, 1e-5\}$  while the momentum value  $\mu$  was fixed to 0.9. The 64-dimensional color histogram (CH)

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

and 225-dimensional block-wise color moments (CM55) provided by NUS-WIDE’s authors and the 64-dimensional CH of Flickr-PTR images are used for comparison.

The proposed model **CNN-PerMLP** is compared to the following personalized tag recommendation methods that use only the users’ preference information and do not consider the visual features: **PITF**, **FM**, most popular tags by users (**MP-u**) [56].

In addition to CNN-PerMLP architecture, several configurations in which we alter the personalized fully-connected layer by a personalized convolutional layer and its position is behind the first and second convolutional layer respectively. These models are named as **CNN-1PerMLP** and **CNN-2PerMLP**. Moreover, an architecture, called as **CNN-2BlockMLP**, with two personalized layers following the first and second convolved layers is also employed for the evaluation.

It is also compared to the non-personalized models including most popular tags(**MP**) [56], the multilabel neural networks (BP-MLLs) [159] that have low-level visual features as the input (**CH-BPMLL**, **CM55-BPMLL**), CNNs utilized for image annotation which optimizes the pairwise ranking loss to learn the parameters as the loss used by Zhang and Zhou [159]. The adjusted models (**CH-PerMLP** and **CM55-PerMLP**) of the proposed model using low-level features were obtained for the comparison. We used the Tagrec framework [67] to learn MP and MP-u, and the Mulan library [141] to learn CH-BPMLL and CM55-BPMLL.

### 6.4.3 Results

Similar to the result in Chapter 5, the MP model, which recommends the global most popular tags to any post, is the worst model. All given images receive the same recommended list of tags which have the highest frequencies of use by all users. Because this set does not connect to any specific user’s historical behaviors or contents of images, the performance of this model is certainly worse than other models, which can catch the relation between tags and other entities as we can see in Figure 6.3. In this figure, the non-personalized

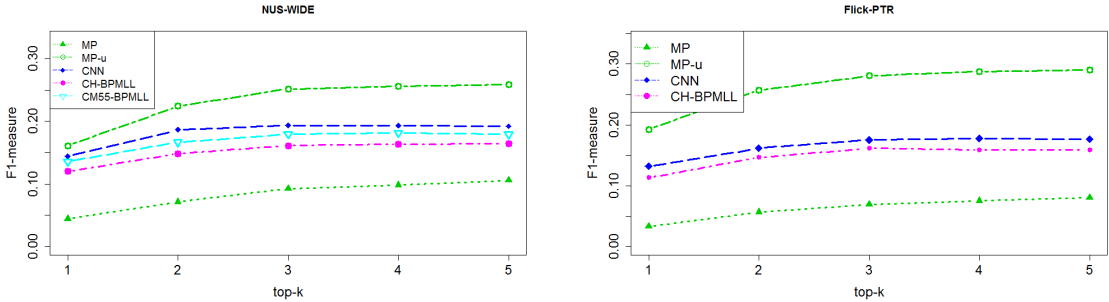


Figure 6.3: Comparison of non-personalized and personalized models

models which are based on visual features of images to predict tags cannot capture the user’s interests and they just recommend tags related to the content. They can capture

common content-based tags used by most users, such as *blue* or *green*, to annotate their resources. So, their recommendation is more relevant to posts than the most popular tags. The assumption is clearly presented in the figure where their result lines go above MP. Clearly, when applying this approach to solve the personalized problem, the prediction quality of these models is lower than that of the personalized model. If we compare between the high-level features learned by a CNN and the low-level features, the model used learnable features captures more information leading to the boosted performance by around 2 percent.

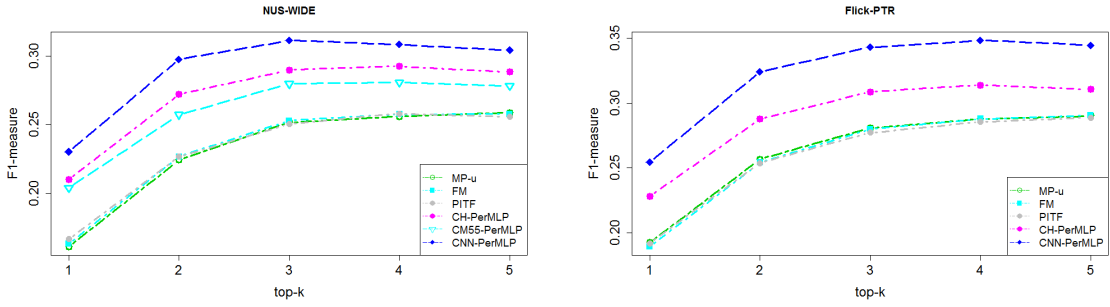


Figure 6.4: Comparison of personalized models

The claim that visual features improve the prediction quality is more evident in Figure 6.4. In the test set having most new images, the weights associated with them are not learned in the training process. So the prediction of personalized content-ignored models like FM and PTF solely depends on users and their results are clearly comparable to the prediction of MP-u. On the other hand, the personalized content-aware models work better than them in this case and recommended tags rely on both users and visual image information. Weights of images in these models are associated with visual features and all images are capably represented by feature vectors. These models will not face the similar issue which the purely personalized models have in the cold-start image scenario. Generally, visual features help to increase the prediction quality around four percent. Similar to the non-personalized models, the supervised features used in the personalized frameworks also prove their strength in the recommendation quality compared to the low-level features. The performance is improved around two to three percent as a result of using the learned visual features.

Figures 6.5 and 6.6 show the efficiency of adding the personalized layer in different levels. If feature maps of the first convolutional layer are personalized, the user's information is lost through several layers. Features achieved by the personalized extractor are overwhelmed by visual elements and they are weaker than features obtained by other configurations. This results of CNN-1PerMLP in these figures prove the above discussion. Although their features enriched by user's information are better than purely visual features to boost up the accuracy, they are still not enough to capture personalized tags. We can also see in Figure 6.7 that personalized feature maps at the first level are not different

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

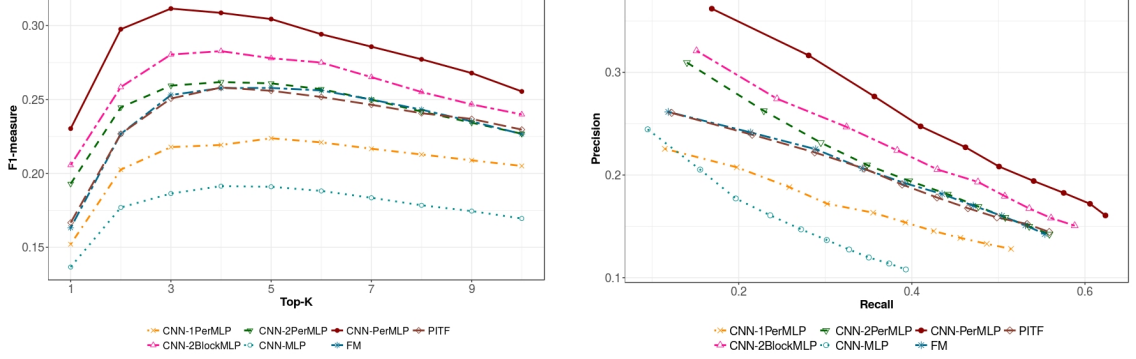


Figure 6.5: The results of personalizing images at different levels for NUS-WIDE-1

obviously.

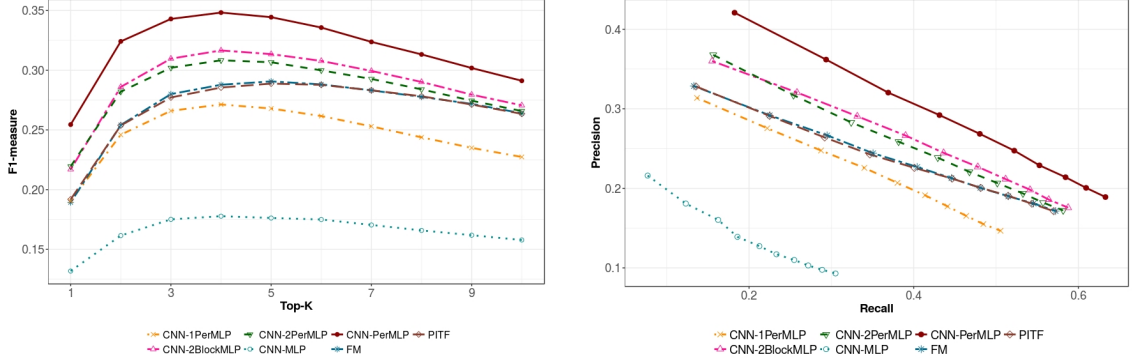


Figure 6.6: The results of personalizing images at different levels for Flickr-PTR

When the personalized layers are shipped forward, the performance of these models is enhanced. In other words, if the user's information is added to features retrieved by the second convolutional layer, the model based on them works better than a model using first personalized features. Feature maps added the user's data at the second level CNN-2PerMLP are more specific to different users compared to the first level as shown in Figure 6.7. When we personalize feature maps of two first convolutional layers, the accuracy of the tag recommendation model is truly enhanced. However, its performance is still far behind the performance of the model based on the personalized fully-connected layer. The model CNN-PerMLP in which visual outputs of the last convolutional layer are aggregated with the user's latent features achieves the best performance.

Examples in Table 6.4 show that the proposed model can predict both personal tags and content tags compared to MP-u that purely predicts personal tags and CNN recommending content tags. As a result, the CNN-PerMLP suggests more relevant tags to the image. For example, in the first photo, the recommender can catch personal tags as *flowers*, *flower*,



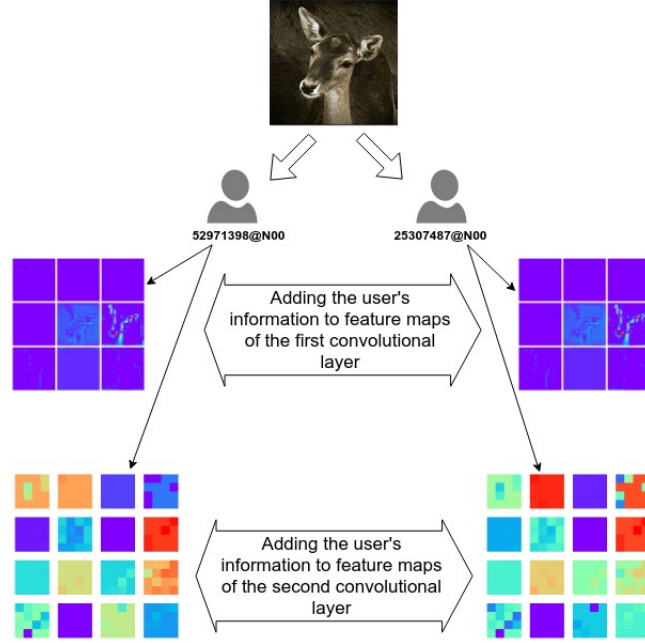




Figure 6.7: Feature maps are different based on users.

*orange* and content tags as *white* or *red*.

Table 6.4: Examples of top 5 recommended tags of CNN-PerMLP, CNN and MP-u

Image	Ground truth	CNN-PerMLP	CNN	MP-u
	flower red orange white	red white flowers orange flower	red woman girl white people	flowers white orange pink flower
	green grass landscape park	landscape green sky park grass	green bravo blue nature flowers	beautiful park landscape color animal

Through Table 6.5 and 6.6, CNN-PerMLP works well in the case that people use their frequent tags or tags related to the image's content to annotate a new image. However, the prediction quality of the model is poor if the users assign tags that are new and do not relate to the content.

## 6. PERSONALIZED DEEP LEARNING FOR TAG RECOMMENDATION

Table 6.5: Examples having the highest accuracy of top 5 recommended tags





Image	Ground truth	Prediction
	lake sunset water blue sun	sunset water lake blue sun
	water man river	river water woman old man

Table 6.6: Examples having the lowest accuracy of top 5 recommended tags

Image	Ground truth	Prediction
	green	sea beach sunset clouds ocean
	people	reflection water light window night

### 6.5 Discussion

In this chapter, we propose a deep learning approach using supervised visual features for personalized image tag recommendation. The experiments show that the proposed method has advantages over the state-of-the-art personalized tag recommendation purely based on tagging history. Moreover, high-level features learned with the predictor have stronger effects compared to low-level features. The additional levels of user data also affect the accuracy of the models in different ways in which adding users to the last layer of the extractor is the best option.

One assumption we can consider to build the neural network based tag recommendation is that we personalize synapses of the network instead of adding a “user bias” to the model. That is like generating  $|U|$  networks for users, but it requires that each network has numerous posts for the training process. If we enable to decompose the large user-based parameter set into smaller subsets, it is feasible to train the model in a limited dataset.

The running time and the lack of data are still issues of the models. To train the proposed models based on a shallow convolutional network, it is required that the number of posts per tags is nearly 1000. If the network has more trainable parameters, this number becomes very large that the current tag recommendation data cannot satisfy. Additionally, most running time of the model is used to learn the extractor although the depth of its network is not large compared to other classification models. It brings us to several questions about whether it is possible to use the knowledge of another very deep classification model, which is learned in the huge domain, to obtain wealthy visual features.

## Chapter 7

# Personalized Tag Recommendation for Images Using Deep Transfer Learning

### Contents

---

<b>7.1</b>	<b>Introduction</b>	<b>63</b>
<b>7.2</b>	<b>Background Review</b>	<b>65</b>
<b>7.3</b>	<b>Problem Extension</b>	<b>66</b>
<b>7.4</b>	<b>The Proposed Architecture</b>	<b>67</b>
7.4.1	Visual Feature Extraction	67
7.4.2	Object Detection	69
7.4.3	Factorization Models	71
7.4.4	Factorization Models for Image-Aware Tag Recommendation	72
7.4.5	Personalized Multilayer Perceptron for Image-Aware Tag Recommendation	73
7.4.6	Optimization	74
<b>7.5</b>	<b>Evaluation</b>	<b>75</b>
7.5.1	Dataset	75
7.5.2	Experimental Setup	76
7.5.3	Results	76
<b>7.6</b>	<b>Discussion</b>	<b>79</b>

---

### 7.1 Introduction

So far, in Chapter 5, we have presented the hybrid approaches based on the factorization and deep learning techniques to predict personalized tags for a given image, while in

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

Chapter 6, we have introduced the CNN architectures to implicitly encode the personalized visual features in estimating the tag’s scores. The similarity of these approaches is the CNN module, which extracts features from the image, while the factorization-based or the multi-layer perceptron module is utilized in predicting the probabilities of tags.

Compared to recent CNN architectures [68, 131, 137], the CNN extractor is shallower in both the connection types and the number of layers. Because of the limitation of the tag recommendation dataset, it is difficult to learn a large network with millions of parameters. Moreover, although we aim to improve the accuracy of the score predictor, most of the time is spent on backpropagating the error through the CNN layers and updating their parameters.

In this chapter, we exploit how to leverage a deeper CNN model to enhance the tag recommendation by using knowledge found in a large scale vision dataset. As a result, most efforts are used to learn the predictor, and it allows us to free the extractor from the learning process in the limited dataset. A deep network is selected as a base network, which is employed as the visual extractor in the tag recommendation model. To overcome the limitation of training data, it learns the extracting skills on a large dataset and uses the knowledge to extract features for images of the smaller dataset.

Besides the visual features, object-based features are also covered in this chapter. The motivation for our approach can be explained easily and follows the way how human beings tag images. Let us have a look at Figure 7.1. The user tagged this image with “urban”, “motorcycle” and “downtown”. While the COCO dataset only allows us to distinguish 80 different objects which are completely unrelated to our task, we can nevertheless detect a person, a motorbike, a car and few further objects. This is also what a human being does and the appearance of the motorbike likely resulted in the tag “motorcycle”. However, object detection can also help to recommend tags such as “urban”. This tag is obviously no object which you see on the image, but the recommender system can learn that whenever motorbikes, cars, and people are detected on an image that an urban area or city is pictured. In a similar way, the classification algorithm can extract image features such as specific shapes, colors and so on. Thus, the combination of these features will be used as the image’s factor such that the performance of the recommender is improved compared to the models that rely on only one kind of features or ignored contents of images.

In Chapter 5, a FM model is modified to receive the visual information as its input instead of the index. It acts as an output layer in the network to predict scores of tags reflecting the user’s preference and the image’s content. The FM-based method in this chapter is a variant of the previously proposed approach, which recommends tags related to the given user, the image content, and the object-based context of the image. A PITF model, which also belongs to the family of latent factor models and shares some characteristics with FM, would be a promising approach to deal with the content-aware tag recommendation. Instead of using only index-based inputs, we propose an approach that allows the model accepting dense values as its input. Furthermore, the object mode is added to the PITF-based model such that the context information is utilized to boost up the model’s performance.

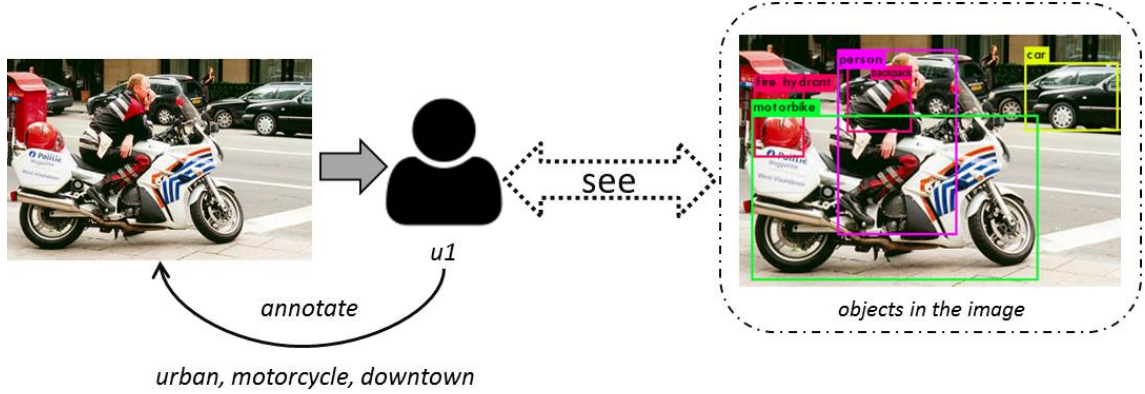


Figure 7.1: When tagging an image, the user is highly influenced by what she sees on the image. In this example, the user chooses “motorcycle” due to its occurrence. Furthermore, the tag “urban” is chosen since the image shows a street, people, cars and other things typical for cities. Our idea is to create an automatic system that uses object detection as a part of it to improve the tag recommendation performance.

Our experiments are conducted on a real-world dataset, namely NUS-WIDE, and we can show that our proposed way of extracting image features improves the accuracy of the tag recommender by at least one percent.

## 7.2 Background Review

Beyond various modeling methods, the usage of context has more become popular for enhancing the performance of the models for different tasks. For example, Gantner *et al.* [34] proposed a context-aware item prediction, which encodes time as one mode to construct a 3-dimensional tensor of interactions among users, items, and time-based contexts. Then, a factorization model was employed to predict scores of items.

Employing the context factor is also considered in order to improve object detection and classification, and many studies have demonstrated its positive influence on the predictor’s performance [11, 32, 44, 52, 54]. The context information is retrieved from the external source with traditional image descriptors, such as geographic coordinates [5, 10, 139], texts from Internet [146], user-provided tags [44].

One popular image task which has drawn the attention of many researchers recently is the image classification [68, 78, 157]. Likewise, object detection is also a task for image understanding and has attracted numerous studies to solve the problem, especially in the deep learning field [36, 37, 38, 110, 112, 113, 137]. While the image classifier predicts one or more labels belonging a category to assign for an image, the object detection is defined as the task of localizing and classifying objects in an image. Because they are obviously related, some research [19, 48] is investigated to combine object detection and image

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

classification. It means that the output of one task is used as a context to improve the performance of the other task. Their experiment results proved that the object detection impressively supports improving the image classification.

In the thesis, we seek to learn the methods to solve the tag recommendation task, which can be seen as a variant of multi-label classification. Because of the interdependence of image classification and tag recommendation, it raises the possibility of using objects detected in an image as a context for our aiming task. However, the difference between our target and the studies [19, 48] is that we will leverage the object detection in a source dataset and use its results as the inputs to promote the tag recommendation, instead of mutual boosting both tasks.

### 7.3 Problem Extension

Similar to previous chapters, the purpose of a personalized tag recommendation is to discover a sorted list of tags reflecting the user's preference and item's content. Digital images appearing in the dataset are composed of the set  $\mathcal{R}$ , which are the input of visual and object-based feature extractors. Generally, the prediction process banked on a given post is formulated as

$$\hat{y}_{u,i} = f(u, R_i; \Theta) = f(x_u, f^I(R_i; \theta), f^O(R_i; \vartheta); \omega) \quad (7.1)$$

where  $f^I$  maps the given digital image to a dense visual representation;  $f^O$  is also an extractor to generate object-based features for the given image.

The above data are organized in the target domain and two source domains in which the extractors learn their extracting skills contain auxiliary datasets  $\mathcal{D}^I$ ,  $\mathcal{D}^O$ . Apparently, both the target data and the source data are regarding images, but their tasks are defined differently; i.e. the target is tag recommendation while the source is image classification or object detection.

Consider the classification task in the source domain containing the set of  $|\mathcal{X}^I|$  images. For a given image  $X_i^I$ , the source model predicts probabilities of classes and assigns the class having the highest probability to the image,

$$g(X_i^I; \theta, \zeta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^{|C_1|} \quad (7.2)$$

where  $|C_1|$  is the number of classes. In this network, the extractor parameterized by  $\theta$  is in charge of generating the visual representation, which is the input of the classifier characterized by  $\zeta$  to estimate probabilities of labels.

The object detection task aims to localize and classify objects existing in an image. Let  $\mathcal{X}^O$  denote the set of images in the detection dataset. For a given image  $X_i^O$ , the source detection model predicts the set of detected objects with their coordinates and labels.

$$g(X_i^O; \vartheta) : \mathbb{R}^{d \times d \times 3} \rightarrow \{C_2 \times \mathbb{R}^4\} \quad (7.3)$$

where  $C_2$  is the set of object categories existing in the source dataset.

The auxiliary models share some or all parameters with the target model such that similar parameters learned in the source models are carried to the target model to raise its performance. The transferred parameters are mostly fixed to the extractors including the object-based or the visual branches. So, our goal is to learn an accurate predictor  $f(\cdot; \omega)$  from  $\mathcal{S}^{train}$  or  $\mathcal{P}^{train}$  such that the distances between observed and unobserved tags are maximized.

## 7.4 The Proposed Architecture

Our personalized image-aware tag recommendation aims at taking benefit of deep learning methods to extract representations of images and improve the recommendation capability of factorization models. In the previous architectures described in Chapter 5 and 6, the visual information of images is mainly used with the user's index to compute scores of tags. The proposed model in this chapter exploits the image-inside context concatenated with the visual features to improve the performance of the tag recommender. Our proposed architecture is illustrated in Figure 7.2.

According to that figure, there are two deep network branches adopted to elicit the image's contents/contexts and a factorization layer acts as an estimator, which computes scores of tags found in the dataset. In addition, other two deep networks having the similar architectures with the extractors are pre-trained in larger datasets to solve different tasks such that their knowledge will support the performance improvement of the target model, the tag recommendation. Indeed, one of them is trained on the ImageNet dataset for the image classification task, and the other is trained on the COCO dataset for the task of object detection. Their parameters are transferred partly or fully to the target and fixed during the training process. The tag recommendation is trained on the target domain to determine parameters of the factorization layer.

### 7.4.1 Visual Feature Extraction

In this approach, we again adopt a CNN to derive a visual representation of an image as Chapter 5 and 6. In the previous architectures, a CNN, which has three convolutional layers interconnected by two max pooling layers in a sequential order, is utilized as the extractor. Compared to vast amounts of CNN architectures for image-related tasks, the architecture is tiny and may miss some contents because of low-resolution inputs.

To achieve richer visual features, we take into account a deeper architecture, which has numerous convolutional layers stacked sequentially, replacing for the shallower extractors in the previous models. One of the state-of-the-CNN approaches for image classification is the VGG network [131]. Its architecture contains multiple convolutional layers located in five sequential blocks and several max pooling layers are alternated between these blocks. The predictor block involves several fully-connected layers to predict the probabilities of different labels. The arrangement of the network's layers is illustrated in Figure 7.3. VGG-based models are not only used to predict image's labels or localize objects in images

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

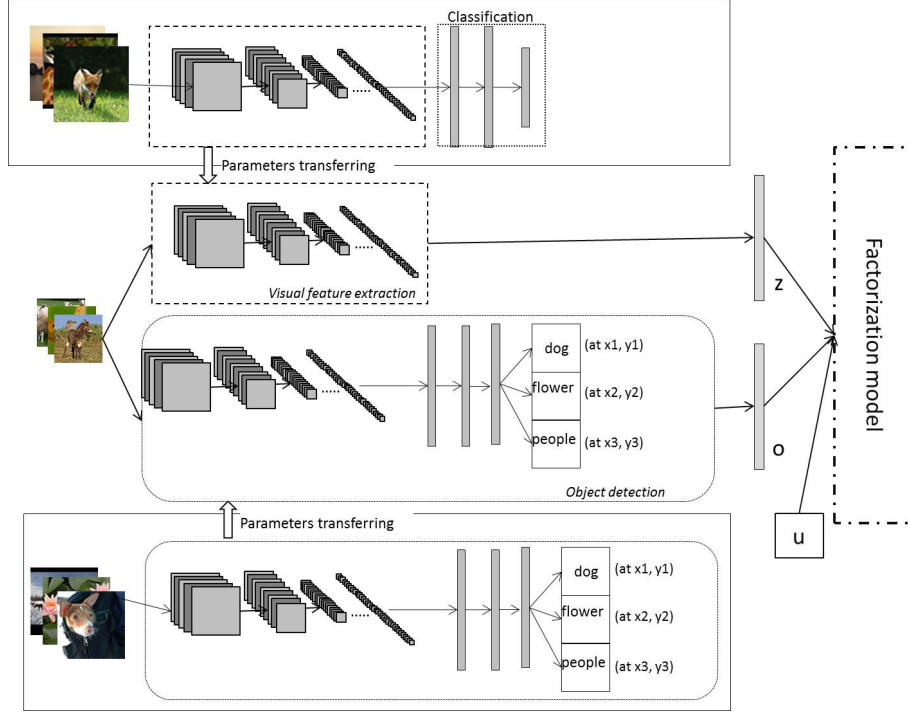


Figure 7.2: The proposed architecture for personalized content-aware tag recommendation. We train one network for the task of image classification and one network for the task of object detection on two different datasets. These networks are finally used to extract image features or detect objects. These features and predictions are used as visual features in order to train a factorization model.

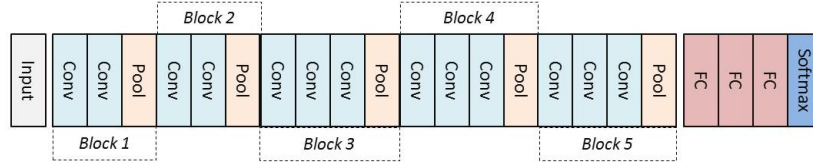


Figure 7.3: The architecture of VGG-16 that having 16 weighted layers

directly but also reused partly as a visual extracting block in a new deep model [81, 82, 99, 102, 114]. Hence, it is a good candidate for the visual branch in the proposed model, which allows a higher resolution image entering the network.

Due to the abundant trainable parameters, the deep network needs to be trained on a large dataset, but we conducted the experiments on a smaller dataset having around less than 80 thousands instances. Moreover, most running time to train the proposed models from scratch is principally spent on updating the extractor's weights while the predictor is our main focus in the approach. As a common practice [37, 49, 81], the pre-trained



weights of the base network on a large image dataset are transferred to the target model to boost up the performance of its extractor.

Among several variants of VGG networks, we picked the 16-layer version, which includes 16 trainable layers described as the D version in [131], as the base network. Firstly, we train the full network on 1000-labels ImageNet dataset for the image classification. Later, all fully connected layers and the softmax output are removed and replaced by a global average pooling to produce a dense output with the length 512. Finally, the weights of the subnetwork are fixed and used as the feature extractor in the tag recommender system. The extracting features process is formulated as:

$$z_i := f_{\text{vgg16}}^I(R_i; \theta) : \mathbb{R}^{224 \times 224 \times 3} \rightarrow \mathbb{R}^m$$

### 7.4.2 Object Detection

Deep learning not only achieves state-of-the-art performance for image classification but also is applied successfully for the task of object detection. In principle, the object detection ascertains the location identified by a rectangle and predicts the class of each object appearing in an image. One of basic models for the deep learning-based object detection is Faster R-CNN [113] promoted from the R-CNN [37] and Fast R-CNN [36]. In the first variant of the R-CNN group, the region proposals based on the selective search algorithm determine possible regions covering objects and a CNN-based network is applied to extract visual features for each region. The representation is fed to a SVM classifier to predict labels of the region and to a linear regressor to optimize the bounding box of the object. Instead of extracting multiple visual representations of images based on region proposals, an upgraded R-CNN, named as Fast R-CNN, is extracting the visual features over the image before finding proposed regions and replacing SVM with a softmax layer. The Faster R-CNN [113] replaces the slow selective search with a Region Proposal Network (RPN) to generate a fixed set of regions, which will go to the RoI pooling layer of the Fast R-CNN to predict labels and bounding boxes.

Another approach, called the YOLO model [110, 111, 112], looks upon the detection as a regression problem, which passes a given image once through the convolutional network to predict probabilities of classes and coordinates of bounding boxes. In the first variant of the model, each image is divided into a  $m \times m$  grid, and for each cell, the network predicts  $N$  bounding boxes with their confident scores, which reflect how likely any object will appear in the boxes. For each box, the model also predicts probabilities of classes assigned for the recognized object.

The second generation of YOLO, YOLOv2, is based on the DarkNet-19 architecture that is described in Table 7.1. The network comprises multiple convolutional layers mostly having  $3 \times 3$  filters, and the number of feature maps is doubled after each pooling step. The authors integrated a variety of techniques into the network, such as adding batch normalization layers, fine-tuning the classification network at the higher resolution in several epochs, etc (see [110] for details). The model outperforms several state-of-the-art deep learning models on the standard detection tasks like PASCAL VOC and MS COCO

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

Table 7.1: YOLOv2 is a fully convolutional network and is based on the Darknet-19 architecture sketched below.

Type	Filters	Size/Stride	Output
Convolutional	32	$3 \times 3$	$224 \times 224$
Maxpool		$2 \times 2/2$	$112 \times 112$
Convolutional	64	$3 \times 3$	$112 \times 112$
Maxpool		$2 \times 2/2$	$56 \times 56$
Convolutional	128	$3 \times 3$	$56 \times 56$
Convolutional	64	$1 \times 1$	$56 \times 56$
Convolutional	128	$3 \times 3$	$56 \times 56$
Maxpool		$2 \times 2/2$	$28 \times 28$
Convolutional	256	$3 \times 3$	$28 \times 28$
Convolutional	128	$1 \times 1$	$28 \times 28$
Convolutional	256	$3 \times 3$	$28 \times 28$
Maxpool		$2 \times 2/2$	$14 \times 14$
Convolutional	512	$3 \times 3$	$14 \times 14$
Convolutional	256	$1 \times 1$	$14 \times 14$
Convolutional	512	$3 \times 3$	$14 \times 14$
Convolutional	256	$1 \times 1$	$14 \times 14$
Convolutional	512	$3 \times 3$	$14 \times 14$
Maxpool		$2 \times 2/2$	$7 \times 7$
Convolutional	1024	$3 \times 3$	$7 \times 7$
Convolutional	512	$1 \times 1$	$7 \times 7$
Convolutional	1024	$3 \times 3$	$7 \times 7$
Convolutional	512	$1 \times 1$	$7 \times 7$
Convolutional	1024	$3 \times 3$	$7 \times 7$
Convolutional	1000	$1 \times 1$	$7 \times 7$
Avgpool		Global	1000
Softmax			

with shorter running time.

To obtain the object-based context information for an image, we adopt the YOLOv2 as the object-based generation. In the general case, the output of a given image is a set of detected objects  $O_i$ , and each object is presented by a specific label  $c$  with its confident probability  $p$ , and 4 values of its bounding box, as  $O_i = \{(c, p, x, y, w, h) | c \in C_2\}$ , where  $C_2$  is the set of object categories, and  $|C_2| = n$  is the number of existed objects. The context of an image is represented by probabilities of objects appearing in the image encoded in a  $n$ -dimensional vector. Because one categorical object possibly appears multiple times with different probabilities, we select the highest value for the object category. Moreover, the information of bounding boxes is not used in the models, and it is passed over during the extraction process.

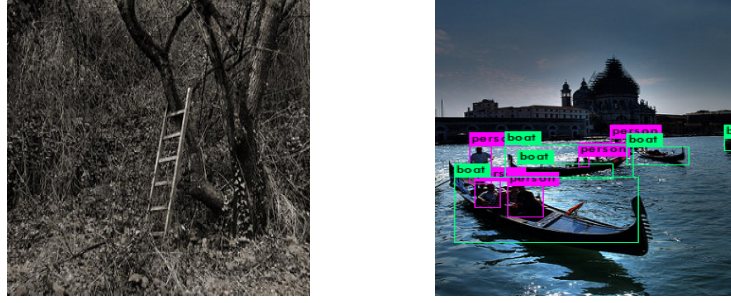


Figure 7.4: While the detector cannot recognize any object in the left image, it detects several objects, such as person or boat, in the right image.

The tag recommendation dataset does not have labeled data for the detection task, so it is necessary to borrow the knowledge from another detection dataset. In reality, we train YOLOv2 on the COCO dataset, which contains 80 categories of objects, to learn the detecting skills. Then, for each image in the target dataset, the network extracts probabilities of 80 categorical objects detected in the image. The values are encoded in a sparse vector representing for the image-inside context depending on the number of objects detected. The process is denoted as,

$$o_i := f_{\text{yolo}}^O(R_i; \vartheta) : \mathbb{R}^{448 \times 448 \times 3} \rightarrow \mathbb{R}^n \quad (7.4)$$

However, unlike the visual extractor where any image is depicted by a visual vector, several images which include the vague difference between objects and the background or unobserved objects, possibly have no object-based context information. One example is shown in Figure 7.4. The extractor is not able to obtain object-based features of the left image which has an unobserved object, the ladder, in the source dataset. On the other hands, for common objects, it comfortably recognizes and localizes them as in the right image.

### 7.4.3 Factorization Models

As mentioned in the above sections, two state-of-the-art factorization models applied widely for tag recommendation are FMs and PITFs that model the interaction between different entities of tag assignments. While the PITF model distinguishes latent features of tags for different pairs of interaction, the FM model shares these features between all pairs of interaction. In more detail, with the input defined as Equation 5.3, the scoring function in FM model is denoted as in the similar way to Equation 5.4.

Because exactly one  $x_u$ ,  $x_i$  and  $x_t$  are one and all others are zero, and we are applying a pair-wise loss function, the prediction function of the FM can be simplified to

$$(\hat{y}_{u,i})_t = w_t + \sum_{j=1}^q (V_{u,j}^U + V_{i,j}^I) V_{t,j}^T \quad (7.5)$$

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

where  $p$  is the number of latent features,  $V^U \in \mathbb{R}^{|U| \times p}$ ,  $V^I \in \mathbb{R}^{|I| \times p}$  and  $V^T \in \mathbb{R}^{|T| \times p}$  are the latent feature matrices of users, images and tags respectively. Similarly, the PITF prediction model is simplified as

$$(\hat{y}_{u,i})_t = \sum_{j=1}^p V_{u,j}^U \cdot V_{t,j}^{T^U} + V_{i,j}^I \cdot V_{t,j}^{T^I} \quad (7.6)$$

where model parameters are denoted as  $V^U \in \mathbb{R}^{|U| \times p}$ ,  $V^I \in \mathbb{R}^{|I| \times p}$ ,  $V^{T^U} \in \mathbb{R}^{|T| \times p}$  and  $V^{T^I} \in \mathbb{R}^{|T| \times p}$ .

With the configuration, the models are plainly based on the relation among different elements, and use the indices of all elements as their input. We cannot directly apply these models to content-aware recommendation where the input contains information of images representing in feature vectors, especially for PITF model.

### 7.4.4 Factorization Models for Image-Aware Tag Recommendation

The aforementioned factorization models focus on using users' preferences, instead of using contents of images. To feed image-based features to the factorization models, the part representing the image in Equation 5.3 is replaced by its image-based features. Depending on the types of image's representation, the input is expanded by the combination of multiple features or is shrunk to the given feature category. For example, the tag recommender consumes the visual features as the image factor of its input,

$$\mathbf{x}_{u,i,t} = \left( \underbrace{\dots, \overbrace{1}^u, \dots}_{|U|}, \underbrace{(z_i)_1, \dots, (z_i)_m}_m, \underbrace{\dots, \overbrace{1}^t, \dots}_{|T|} \right)$$

If the concatenation of visual and object-based features is used to characterize the given image, the input is extended as:

$$\mathbf{x}_{u,i,t} = \left( \underbrace{\dots, \overbrace{1}^u, \dots}_{|U|}, \underbrace{(z_i)_1, \dots, (z_i)_m}_m, \underbrace{(o_i)_1, \dots, (o_i)_n}_n, \underbrace{\dots, \overbrace{1}^t, \dots}_{|T|} \right)$$

With the advantage of the feature engineering, the FM-based model allows changing, removing or adding features to the input. However, the PITF model was originally created to work with one-hot features, so it is modified to work with the content-aware tag recommendation task. Based on the description of the input, we propose different factorization models based on FM and PITF to generate the scoring functions.

If both types of features are used to predict the relevant tags, the scoring functions are formulated as:

- The FM-based formula is

$$(\hat{y}_{u,i})_t = w_t + \sum_{j=1}^p \left( V_{u,j}^U + \sum_{a=1}^m (z_i)_a \cdot V_{a,j}^I + \sum_{a=1}^n (o_i)_a \cdot V_{a,j}^O \right) V_{t,j}^T \quad (7.7)$$

where  $w \in \mathbb{R}^{|T|}$  is the weights of tags;  $V^U \in \mathbb{R}^{|U| \times p}$ ,  $V^I \in \mathbb{R}^{m \times p}$ ,  $V^O \in \mathbb{R}^{n \times p}$ ,  $V^T \in \mathbb{R}^{|T| \times p}$  are the factorized parameters associated with the user, visual, object, and tag factor respectively.

- The PITF-based function is

$$(\hat{y}_{u,i})_t = w_t + \sum_{j=1}^p V_{u,j}^U \cdot V_{t,j}^{T^U} + \left( \sum_{a=1}^m (z_i)_a \cdot V_{a,j}^I \right) V_{t,j}^{T^I} + \left( \sum_{a=1}^n (o_i)_a \cdot V_{a,j}^O \right) V_{t,j}^{T^O} \quad (7.8)$$

where the factorized weights of the tag factor are separated for different interactions  $V^{T^U}$ ,  $V^{T^I}$ ,  $V^{T^O}$ , and all matrices have the same dimension  $V^{T^*} \in \mathbb{R}^{|T| \times p}$ .

These prediction functions are easily changed to accommodate different types of image's properties by removing the addends and parameters associated with the unused features. For example, if the PITF-based predictor only receives the visual features as the image representation, the Equation 7.8 is adjusted as

$$(\hat{y}_{u,i})_t = w_t + \sum_{j=1}^p V_{u,j}^U \cdot V_{t,j}^{T^U} + \left( \sum_{a=1}^m (z_i)_a \cdot V_{a,j}^I \right) V_{t,j}^{T^I} \quad (7.9)$$

Depending on the types of image-based features and the scoring function, the models are named differently. In detail, we can denote them as

- Based on the object-based features: **FM-OD** for the FM-based model and **PITF-OD** for the PITF-based
- Based on the visual features: **FM-IC** and **PITF-IC**
- Based on the combined features: **FM-IC-OD** and **PITF-IC-OD**

#### 7.4.5 Personalized Multilayer Perceptron for Image-Aware Tag Recommendation

For the comparison purpose, we implement a neural network **CNN-PerMLP** as the predictor replacing the factorization. However, the object-based data of images are quite sparse depending on the number of objects detected in an image. In Chapter 5, the neural network-based predictors do not perform as expected for the sparse-dense concatenated features. For this reason, we only compare the proposed approaches with the MLP-based model in regard to using visual features.

We reuse the layer described in Equation 6.10 to personalize each visual elements by adding the user bias to the weighted features, as

$$(h_{u,i})_j = b_j + (z_i)_j \cdot w_j + V_{u,j} \quad (7.10)$$

where  $b \in \mathbb{R}^m$  is the bias term;  $w \in \mathbb{R}^m$  is the weights of the visual factor;  $V \in \mathbb{R}^{|U| \times m}$  is the user bias term. The output of the layer is a vector having the same dimension as the visual vector,  $h \in \mathbb{R}^m$ .

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

The two-layer MLP is used to compute scores of tags based on the generated features,

$$\hat{y}_{u,i} = f(h_{u,i}; \omega) : \mathbb{R}^m \rightarrow \mathbb{R}^{|T|} \quad (7.11)$$

We also use the ReLU activation function to transform the linear sum of weighted inputs, and do not apply the softmax operator in the output layer.

### 7.4.6 Optimization

For the factorization-based models, we also adopt BPR criterion to find the parameters such that the difference between the relevant and irrelevant tags are maximal. The stochastic gradient descent applied to BPR is in respect of quadruples  $(u, i, t^+, t^-)$ ; i.e, for each  $(u, i, t^+) \in S_{train}$  and an unobserved tag of the post  $p_{u,i}$  drawn at random  $t^-$ , the loss is computed and is used to update the model's parameters.

$$\text{BPR}(u, i, t^+, t^-) := \ln \sigma \left( (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-} \right) \quad (7.12)$$

where  $\sigma$  is the sigmoid function. For later use, the difference between two tags is denoted as

$$(\hat{y}_{u,i})_{t^+, t^-} = (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-}$$

The learning algorithm is described in Algorithm 3. For each random post, the given digital image is passed to the extractors to obtain the visual and object-based representations. Besides, a relevant tag and an irrelevant tag are sampled, and the scores of these tags are computed. Later, the gradients of the cost function in Equation 7.12 with respect to the model's parameters are obtained as follows:

$$\frac{\partial \text{BPR}}{\partial \omega} = \underbrace{\frac{e^{-(\hat{y}_{u,i})_{t^+, t^-}}}{1 + e^{-(\hat{y}_{u,i})_{t^+, t^-}}}}_{\psi_{u,i,t^+,t^-}} \times \left( \frac{\partial (\hat{y}_{u,i})_{t^+}}{\partial \omega} - \frac{\partial (\hat{y}_{u,i})_{t^-}}{\partial \omega} \right) \quad (7.13)$$

In order to learn the model, the derivatives  $\frac{\partial (\hat{y}_{u,i})_{t^+}}{\partial \omega}$  and  $\frac{\partial (\hat{y}_{u,i})_{t^-}}{\partial \omega}$  have to be computed. Several parameters rely on one score value, but numerous variables contribute to both scores. Depending on the connection between the parameters and score values, the gradients of the cost with respect of the parameters contain one or both the above derivatives. For examples, from Equation (7.7), the gradients of the loss with respect to parameters of the positive tag are found by

$$\frac{\partial \text{BPR}}{\partial V_{t^+,j}^T} = \psi_{u,i,t^+,t^-} \left( V_{u,j}^U + \sum_{a=1}^m (z_i)_a \cdot V_{a,j}^I + \sum_{a=1}^n (o_i)_a \cdot V_{a,j}^O \right)$$

On the other hand, the gradients with respect to user's parameters are denoted as

$$\frac{\partial \text{BPR}}{\partial V_{u,j}^T} = \psi_{u,i,t^+,t^-} \left( V_{t^+,j}^T - V_{t^-,j}^T \right)$$

**Algorithm 3** Learning BPR based on the SGD with respect to quadruples

---

```

1: Input:  $\mathcal{P}_{S^{train}}, \mathcal{S}^{train}, \theta, \vartheta, \alpha$ 
2: Output:  $\omega$ 

3: Initialize  $\omega \leftarrow \mathcal{N}(0, 0.01)$ 
4: repeat
5:   Pick  $p_{u,i} \in \mathcal{P}_{S^{train}}$  randomly
6:    $R_i \leftarrow \text{resize}(R_i)$ 
7:    $z_i \leftarrow f_{vgg}(R_i; \theta)$ 
8:    $o_i \leftarrow f_{yolo}(R_i; \vartheta)$ 
9:   Get  $t_{u,i}^+ \in T$  whereas  $(u, i, t) \in \mathcal{S}^{train}$ 
10:  Sample  $t_{u,i}^- \in T$  whereas  $(u, i, t) \notin \mathcal{S}^{train}$ 
11:   $(\hat{y}_{u,i})_{t^+} \leftarrow f(u, i; \omega)_{t^+}$ 
12:   $(\hat{y}_{u,i})_{t^-} \leftarrow f(u, i; \omega)_{t^-}$ 
13:   $\text{BPR}(u, i, t^+, t^-) \leftarrow \ln \sigma \left( (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-} \right)$ 
14:   $\Delta_\omega \leftarrow \frac{\partial \text{BPR}(u, i, t^+, t^-)}{\partial \omega}$ 
15:  Update  $\omega \leftarrow \omega + \alpha \Delta_\omega$ 
16: until convergence
17: return  $\omega$ 

```

---

The gradients with respect to the image parameters are defined by

$$\frac{\partial \text{BPR}}{\partial V_{a,j}^I} = \psi_{u,i,t^+,t^-} \cdot (z_i)_a \cdot \left( V_{t^+,j}^T - V_{t^-,j}^T \right)$$

For the MLP-based predictor, we follow the protocol described in Chapter 6. Indeed, we applied the SGD with respect to posts, in which a set of negative tags are sampled instead of selecting one irrelevant tag. The cost function is used as Equation 5.17 to measure the difference between all relevant tags of the post and a subset of irrelevant tags. The backpropagation algorithm is applied to distribute the error back through the predictor’s layers. Because the extractors’ parameters are fixed, the error is not transferred backward to the convolutional layers.

## 7.5 Evaluation

### 7.5.1 Dataset

We conducted experiments on the NUS-WIDE-1 and NUS-WIDE-2 datasets implemented in Chapter 4. The color images used to extract visual features were rescaled into  $224 \times 224$  dimension while to obtain object-based features, they were resized into  $448 \times 448$  pixels. The distribution of posts per tag in NUS-WIDE-1 is more balanced than in NUS-WIDE-2 which has more than 50 percent of tags appearing less than 500 times.

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

### 7.5.2 Experimental Setup

The visual features extracted are combined in a 512-dimensional vector while the object recognition probabilities of a given image are appended in an 80-dimensional vector. The factor dimension for both factorization architecture is fixed in 128. For the MLP-based model, the number of hidden units are also set to 128. The evaluation metric used in this chapter is the F1-measure in top K tag lists as described in Equation 5.24.

The best learning rate  $\alpha$  are searched within the range  $\{10^{-2}, 10^{-3}, 10^{-4}\}$  and the best L2-regularization  $\lambda$  are found from the range  $\{10^{-5}, 10^{-6}, 10^{-7}\}$ . The proposed models **FM-IC-OD** and **PITF-IC-OD** are compared to following personalized tag recommendation approaches that are based only on the users' preference: **PITF** and **FM**. Moreover, these models are also compared to the factorization models using visual features or object detection features: **FM-OD**, **PITF-OD**, **FM-IC** and **PITF-IC**. We evaluate the proposed approaches with the MLP-based models proposed in the previous chapter: **CNN-PerMLP**. Through Chapter 5 and Chapter 6, the non-personalized CNN-based models achieves the worse performance than the personalized models, so we will not evaluate the models in this chapter.

### 7.5.3 Results

As shown in Figure 7.5 and Figure 7.6, we make the observation that the personalized models **FM** and **PITF**, which do not consider content information, have the worst performance. They solely depend on the users' preferences and their power in catching the interaction between new images with other elements is not effective. In the NUS-WIDE dataset, most images in the test set do not appear in the training set and their latent parameters are not learned.

The claim that image-features improve the prediction quality is again shown in these figures. These features boost the performance from one percent to more than three percent. Despite improving the model performance substantially, the object-detected features, which sparsely represent an image by a group of specific objects, are less effective than visual representation. Among personalized content-aware models, the models **FM-OD** and **PITF-OD** achieve the lowest accuracies although they perform better than FM or PITF. One reason may be that the extractor is unable to depict a few images; i.e, they are represented by zero vectors. In addition, images comprising the similar objects can be represented by similar feature vectors although they have different contexts. For example, zebras appear in both the zoo's and wildlife's pictures with different numbers, as in Figure 7.7. With the object-based approach, the number of objects is not counted on to generate the image's features such that the extractor cannot abstract the zoo and wildlife from the number of animals. Moreover, because the most popular tags in these datasets are related to color such as blue or green, the object detection cannot capture this information and the models using them will miss these tags.

Otherwise, the visual features capture more unique information of a given image which is able to be represented by a dense vector. The color information is also encoded by the



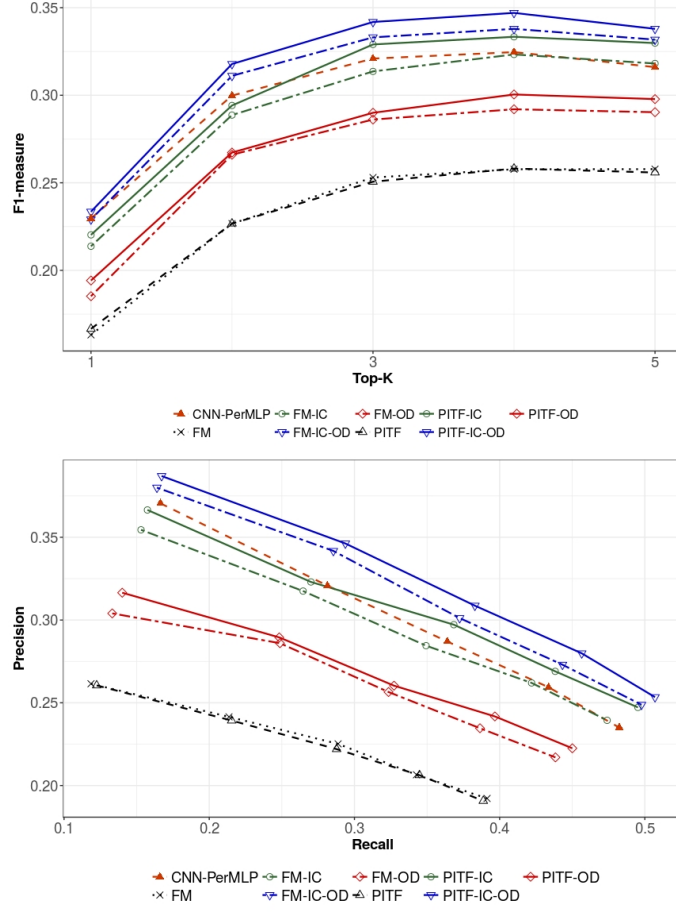


Figure 7.5: F1-measure and Precision-Recall for NUS-WIDE-1

extractor in feature vectors, so compared to object-based features, visual features are quite richer. Not surprisingly, the performance of the models, **FM-IC** and **PITF-IC**, using visual features is better than the models using only the information of object detection. The combination of image-based features prove the powerful abilities in boosting performance. They can capture general object information and unique visual features of a given image. So these features are richer than the others, and the accuracy of the **FM-IC-OD** and **PITF-IC-OD** model provide the best results.

Moreover, the PITF-based models generally work better than the FM-based in most cases. They separate the latent features of tags depending on the elements that they interact with. So they can capture the different representations of tags and combine the scores computing for each interaction into the final score. The difference between the PITF-based and FM-based approaches is clearly in the models using visual or object features; i.e, the FM-based results are always under the PITF-based results.

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

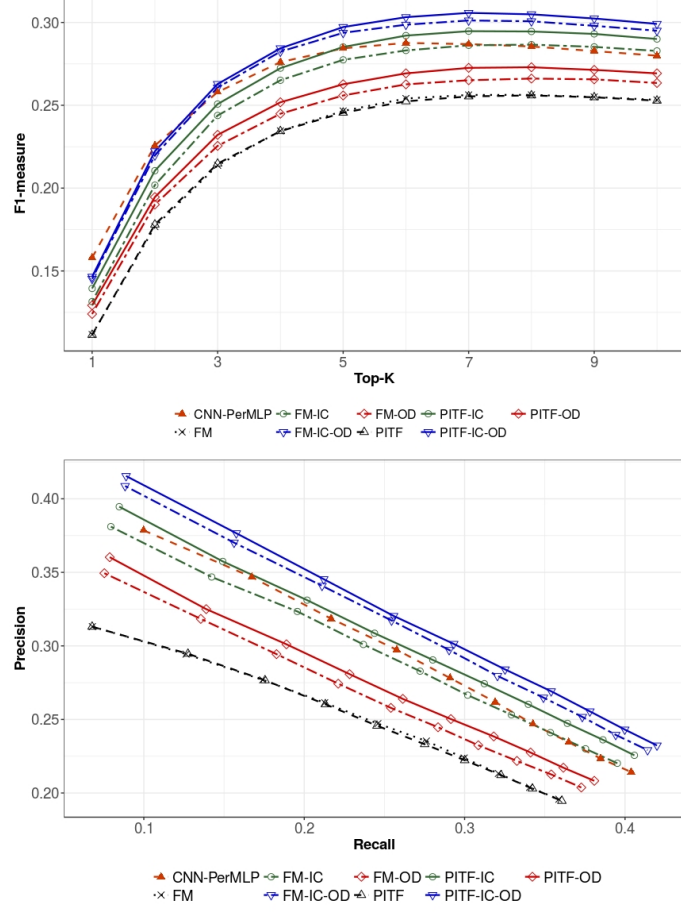


Figure 7.6: F1-measure and Precision-Recall for NUS-WIDE-2

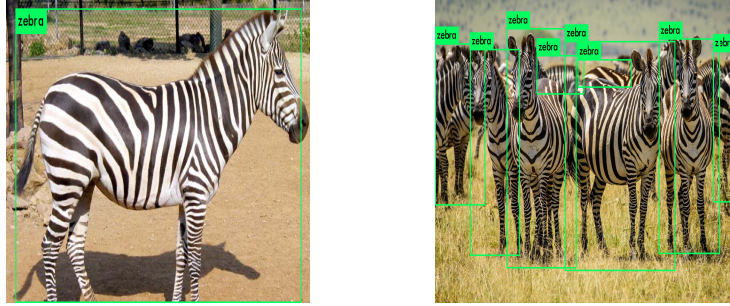




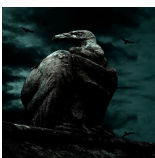
Figure 7.7: For the same object "zebra", while the left image is tagged by "zoo", the "wildlife" tag is assigned for the right image.

When the content-aware models are enriched by the context information, they achieve

superior results compared to their family models; i.e, the FM-IC-OD using both visual and object-based features are better than FM-IC using only visual description. Similar to the visual case, the FM-based model still loses the PITF-models on F1-measure. Interestingly, the MLP-based model based on the personalized visual features, **CNN-PerMLP** outperforms the FM-IC and is even comparable to FM-IC-OD in the top-1 evaluation. However, it drops out if we measure the accuracies in the top-5 set; i.e, at top-5, its performance is equal to the FM-IC.

Examples in Table 7.2 show that the proposed models can capture the visual-based tags and object-tags compared to the models that are purely based on one type of image features. For example, **FM-IC-OD** recommends to a given user the object-based tag as “bird” and the visual-based tag as “black” in the third images.

Table 7.2: Examples top recommended tags of factorization models using different types of features

Image	Ground truth	FM-OD	FM-IC	FM-IC-OD
	wildlife cute squirrel	animal wildlife cat eating squirrel	cat pet dog wildlife lion	wildlife squirrel animal cat eating
	mountain sheep	sheep germany landscape field deutschland	germany deutschland landscape green england	sheep germany landscape field france
	clouds gothic stone bird dark castle wall	bird sun sky water tree blue silhouette beautiful prey sunset	wales cloud water black fresh silhouette sun fab mountain waterfall	bird water lens black white sun aqua sky fab wales

## 7.6 Discussion

In this chapter, considerable attention has been focused on the question of whether the image-inside context supports improving the image’s feature quality used in tag recom-

## 7. PERSONALIZED TAG RECOMMENDATION FOR IMAGES USING DEEP TRANSFER LEARNING

---

mendation. We argue that the context representations, known as object-based features, strengthen the performance of the tag recommendation when they are used along with the user's tagging information or combined with the visual features. Moreover, we showed how to use pre-trained knowledge to boost up the performance of the tag recommendation. The extension of the factorization models for tag recommendations is able to recommend tags related to objects in images, tags representing visual attributes and tags which are typically chosen by a user. The experiments show that different types of image-based features increase the accuracy of tag recommendation on different levels. Additionally, an open issue is that the image-outside contexts, such as text, would also improve the prediction results, and compared to object-based features, how the text-aware tag recommendation performs. We will introduce this approach in the next chapter.

## Chapter 8

# Personalized Tag Recommendation Using Text Transfer Learning

### Contents

---

<b>8.1</b>	<b>Related Work . . . . .</b>	<b>82</b>
<b>8.2</b>	<b>Problem Extension . . . . .</b>	<b>83</b>
<b>8.3</b>	<b>The Proposed Approach . . . . .</b>	<b>84</b>
8.3.1	Visual Feature Extraction . . . . .	85
8.3.2	Textual Feature Extraction . . . . .	87
8.3.3	Object-Detected Feature Extraction . . . . .	91
8.3.4	Factorization Models . . . . .	92
8.3.5	Factorization Machine with Tag Features . . . . .	93
8.3.6	Personalized Neural Networks . . . . .	93
8.3.7	Optimization . . . . .	98
<b>8.4</b>	<b>Evaluation . . . . .</b>	<b>98</b>
8.4.1	Dataset . . . . .	98
8.4.2	Experimental Setup . . . . .	99
8.4.3	Results . . . . .	100
<b>8.5</b>	<b>Discussion . . . . .</b>	<b>104</b>

---

In the previous chapter, we have shown that good results can be achieved by casting the implicit image context, the object-based representations of images, to predict scores of tags. Moreover, to save running time for the predictor, transfer learning techniques are utilized to supply the valuable knowledge of a senior deep network for the target model. Comparing to a shallow network trained in the limited dataset, the deep extractor with the pre-trained knowledge is capable to provide richer features such that the performance of the latent factor models are impressively boosted up.

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

In practice, the external information, such as the text source, also provides the fruitful support to improve the performance of the predictor. Although there are noisy words included in the image’s descriptions, such as camera meta-data or Flickr groups, and images do not always contain textual information, tokens extracted from images reflect the “dictionary” of users and the semantic connection to tags. For this reason, textual features retrieved from available tokens are considered to append with visual features such that image features can represent both the visual content of the image and its descriptive words. In this chapter, we propose incorporating the word embedding technique to retrieve the textual context into the latent factor models and the neural networks. Moreover, instead of learning embeddings from scratch in a small target dataset, we continue using the transfer learning techniques to learn embeddings of tokens in a large corpus and reuse the generated dictionary in the tag recommendation models.

### 8.1 Related Work

Different contexts have been examined in many studies of tag recommendation. In [35], both the tagging history and the text-based global information, a variant of the tf-idf features, are compromised to construct tag recommendation. The hand-crafted visual features are integrated with time and geolocation as three searching factor in [106]. For a new image, its neighboring images are selected in terms of visual features, geographical coordinates, and the image taken time. These candidate images contribute their tags with voting scores to the center decision so that the most frequent tags are chosen to assign to the image.

Liu *et al.* [80] continue mining the geographical information with the textual and visual resources. The authors construct the user-specific space for the visual and textual features, and the geo-specific space towards tags. For a new image, the candidate tags based on the user and the geographic information are found in the subspaces. These tags are used as keywords to search similar images from the photo set, and the visually similar photos are kept to retrieve the votes for relevant tags. The neighboring voting approach based on the visual context, textual meta-data, and user’s preference is also used by Shah *et al.* [128]. In their method, they group users based on the tagging behavior, generate lists of candidate tags with their scores based on co-occurrence information and votes of neighbors, and apply the random walk to find the final list.

In [109], the context information comes from the geo-coordinates and image taken time. A deep network, **ConTagNet** with two components is employed to concatenate visual and contextual features, and predict multi-labels for an image. While the CNN extracts visual features from image pixels, the NN provides latent features of the context. We can consider the tag-based information as one contextual factor to improve the predictor. In [147], the visual representation and label embeddings are joined in one network. The semantic representation of a label is obtained by a Long Short-Term Memory Network (LSTM) [53, 103], whereas a CNN acts as the visual extractor. These features are combined to make a visual-semantic representation by a projection layer, and then the probabilities of

labels are computed based on it.

The context we used in the proposed approaches is the textual information, which is represented by a dense vector. These features are concatenated with visual features generated by a CNN-based extractor to be passed to a latent factor model. The time and geographical information are temporarily ignored in the current work, but they are the next step in our future research. In other approaches, we come up with predicting scores of tags individually with respect to the image content and the textual context and aggregating these scores to find tags with the best scores.

## 8.2 Problem Extension

In Chapter 7, we have formulated the recommending tag problem in terms of two representations of an image, visual and object-detected features. In this chapter, we are going to propose an extended approach which attains textual features combined with visual ones in a content-aware scenario. Indeed, when compared to previous methods, the difference here is the additional steps which obtain textual features of an image. These features are exploited from the image's descriptions and titles that can be viewed as strings of characters. Similar to the previous chapter, the proposed models also rely on the ternary relationships  $\mathcal{S}$  among users  $U$ , images  $I$ , and tags  $T$ ; additionally, the set of *posts* which are pairs of users and images is denoted as  $\mathcal{P}$ .

The pixel values of an image  $i$  are comprised in a tensor  $R_i^{vis} \in \mathbb{R}^{d \times d \times 3}$  that is used to extract a visual vector  $z_i^{vis} \in \mathbb{R}^m$ . In addition to the RGB information of the image's pixels, an image may be described by a sequence of characters, which are elicited from its description and title. The cleaning step is applied to remove noise, such as misspelling words or the hyperlinks, from the string of characters such that an image will be delineated by a set of tokens/words with their frequencies

$$R_i^{txt} = \{(w_i^{txt}, n_{w_i^{txt}}) | w_i^{txt} \in D \wedge n_{w_i^{txt}} \in \mathbb{N}^+\}$$

where  $D$  is the vocabulary set of the dataset. We can encode the textual data of the image as a sparse  $|D|$ -dimensional vector which has only  $|R_i^{txt}|$  nonzero elements. The highly sparse representation will be transformed to a smaller dense vector  $z_i^{txt} \in \mathbb{R}^n$  capturing the semantic correlation among words.

We also seek to learn a machine learning model

$$\hat{y}_{u,i} = f(u, R_i^{vis}, R_i^{txt}; \Theta) : U \times \mathbb{R}^{d \times d \times 3} \times \mathbb{R}^{|D|} \rightarrow \mathbb{R}^{|T|} \quad (8.1)$$

where  $\Theta$  is the parameter set of the model. In a similar way to the previous chapter, knowledge in the source tasks is transferred to the target so that the performance of the tag recommendation model is improved in both the accuracy and the training time aspects. To be more precise, the visual extractor

$$z_i^{ic} = f^{ic}(R_i^{vis}; \theta) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^m \quad (8.2)$$

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

gains the extracting skill  $\theta$  from a classification model

$$f^{Sc}(\theta, \gamma) : \mathbb{R}^{d \times d \times 3} \rightarrow \mathbb{R}^C \quad (8.3)$$

where  $C$  is the number of classes, to produce the visual representation  $z_i^{ic}$ .

Likewise, the textual extractor

$$z_i^{we} = f^{we}(R_i^{txt}, \vartheta) : \mathbb{R}^{|D|} \rightarrow \mathbb{R}^n \quad (8.4)$$

maps the expected words into the pre-learned dictionary  $\vartheta \in \mathbb{R}^{|D| \times n}$ , which is achieved from a source word-embedding model, to generate a dense semantic representation.

### 8.3 The Proposed Approach

The proposed tag recommendation aims taking benefit of deep learning methods to extract visual information and word embedding techniques to retrieve textual information so that the recommendation based on the factorization models achieves better performance. The architecture is illustrated in Figure 8.1. The predictors are not limited to the factorized

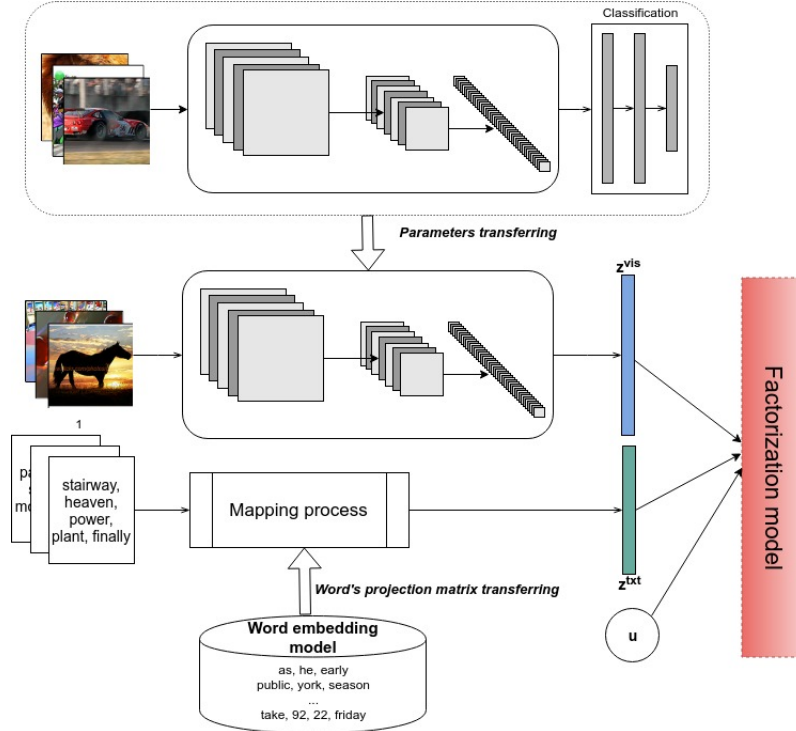


Figure 8.1: The architecture of a content-aware tag recommendation based on the visual and textual features of images.

models, but are possibly replaced by a personalized multi-layer perception.



The proposed hypothesis can be also viewed as a composition of two functions: the feature extractor  $f^e$  and the score estimator  $f^p$ . In particular, the extractor is formed of two sub-functions, the visual extractor, and the textual mapping function. We can denote the process as

$$\begin{aligned}\hat{y}_{u,i} &= f(u, R_i^{vis}, R_i^{txt}; \Theta) = f^p\left(u, f^e(R_i^{vis}, R_i^{txt}; \theta, \vartheta); \omega\right) \\ &= f^p\left(u, f^{ic}(R_i^{vis}; \theta), f^{we}(R_i^{txt}; \vartheta); \omega\right)\end{aligned}\quad (8.5)$$

where  $f^{ic}$  derives the visual features  $z_i^{ic}$  of the image  $i$  and  $f^{we}$  produces the average semantic values of textual features  $z_i^{we}$  for this image. In addition,  $\theta$ ,  $\vartheta$  and  $\omega$  are parameters of these extractors and the predictors, which can be learned during the training phase or retrieved from the “teacher” models. Additionally,  $\theta$  and  $\vartheta$  are knowledge received from the other models, so they will not be updated in the training process.

#### 8.3.1 Visual Feature Extraction

As the previous models, the visual extractor also inherits the extracting knowledge from a source deep CNN model, which is built to solve the classification task. In general, a deep CNN model for classification is composed of several blocks of stacked convolutional layers and a few fully-connected layers close the deep network with the probabilities of labels. An example of this stacking type is the VGG [131], one of the state-of-the-art architectures in image classification, containing five sequential convolutional blocks interleaved by max pooling layers. For the classification task, a multilayer perceptron network having two hidden layers and one softmax output layer is employed to predict the classes’ probabilities. Two main variants of the VGG depend on the number of weighted layers including the extractor and the predictor blocks: VGG-16 with 16 weighted layers and VGG-19 having 19 parameterized layers. The architectures of these networks are illustrated in Figure 8.2 and Figure 8.3.

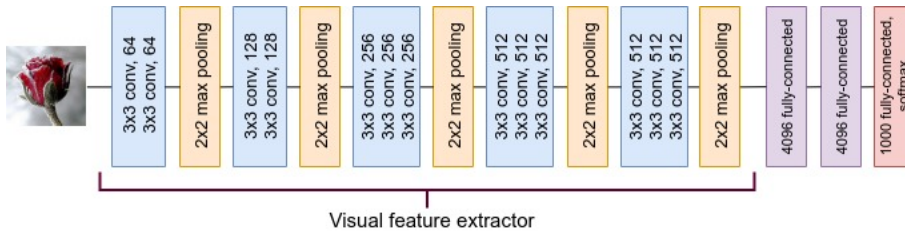


Figure 8.2: The architecture of VGG-16 that has 16 weighted layers

Besides the subsequent connections between layers or blocks, He *et al.* [50] proposed *shortcut connections* in a deep residual learning model known as ResNet. Assume the expected output of a stacked block as  $\mathcal{H}(x)$  and the stacked mapping of the block can be

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

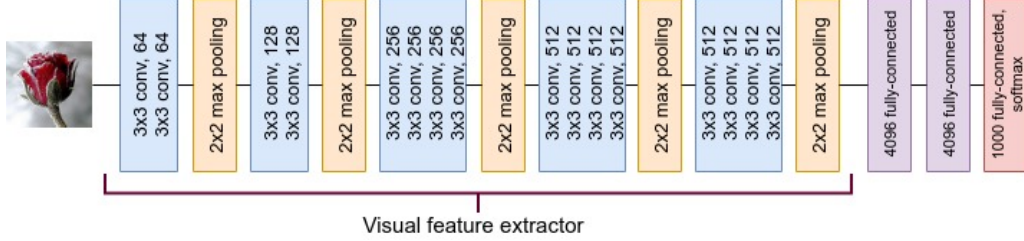


Figure 8.3: The architecture of VGG-19 that has 19 weighted layers

reformed as a residual function of the input and the output of this block  $\mathcal{F}(x) := \mathcal{H}(x) - x$ . The desired output can be approximated based on the residual and the input as  $\mathcal{F}(x) + x$ . Similar to VGG models, a ResNet can be relatively seen as a combination of five convolutional blocks and one fully-connected output layer. However, it contains several residual sub-blocks inside each component as illustrated in the Figure 8.4. Depending on the num-

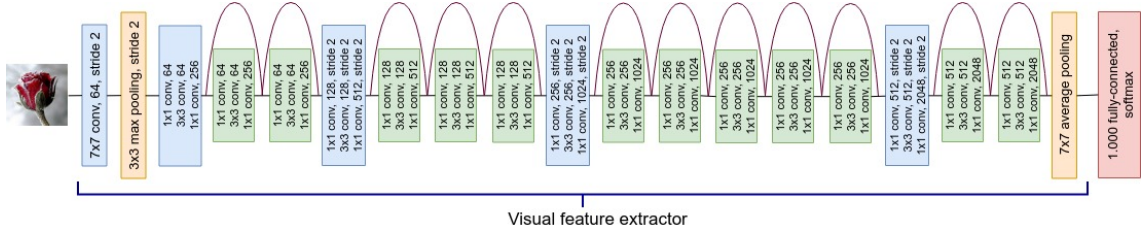


Figure 8.4: The architecture of ResNet-50 that has 50 weighted layers

ber of trainable layers, the depth of the network is different. In the proposed approach, we apply the convolution block of a ResNet-50, which consists of 50 trainable layers including the convolutional and full-connected layers, as a feature extractor. Because of the large number of parameters, these models are required to be trained on sizeable labeled data which are the limitation of the dataset used to conduct the tag recommendation experiments. In addition, the running time of these deep models is painful so that we mostly concentrate on learning the extractor.

As in the previous chapter, the deep transfer learning technique is employed to improve the performance of the extractor. Firstly, a deep network, such as VGG-16 or ResNet-50, is trained on the ImageNet dataset for the image classification. Then, the classifier, a stack of fully-connected layers, is replaced by a global average pooling to extract visual features for the new task, the tag recommendation task. The constructed extractor is fixed with the transferred knowledge when it is adapted for dissolving the personalized tag recommendation. We denote the extraction process as

$$z_i^{ic} := f_{vgg16/vgg19/resnet50}^{ic}(R_i^{vis}) : \mathbb{R}^{224 \times 224 \times 3} \rightarrow \mathbb{R}^m \quad (8.6)$$

where  $m$  is the feature size;  $m = 512$  if the extractor is a VGG-based architecture while  $m = 2048$  if the ResNet-50 extractor is used.

### 8.3.2 Textual Feature Extraction

A valuable resource which the tag recommender probably relies on to estimate tags is the textual source of an image including its title and description. In the raw form, an image is formed in a sequence of characters which people can easily read and understand. However, for purpose of using its information to predict tags, the discrete, categorical data are encoded into a numerical format.

One of the most simple, fundamental techniques to obtain document representation is term frequency-inverse document frequency (tf-idf) [120, 124, 132, 153]. The representation of a document contains weights of terms or words which measure the relative importance of terms in the document. Scores of words in a document are computed by multiplying two factors including frequencies of terms (tf) and the inverse frequency of the given document (idf). In fact, the frequency of a word  $w$  in a document  $d$ , which is the number of times the word appears in that document, measures how important the word is to the document. The  $tf$  score is commonly computed by normalizing the counting value in the document as

$$tf(w, d) = \frac{n_{w,d}}{|d|}$$

where  $n_{w,d}$  is the number of times  $w$  appears in  $d$ ;  $|d|$  is the length of the document or the number of words appearing in the document. To reduce the influence of meaningless, common words, such as stop-words, the  $idf$  scores are multiplied to the  $tf$  scores to diminish the weight of repeated words and gain the weight of rare words in the corpus. Unlike the  $tf$  scores calculated on a specific document, the  $idf$  score of a word  $w$  is computed on the entire document set and it is defined as

$$idf(w, D) = \log \frac{|D|}{n^w}$$

where  $|D|$  is the total number of documents in the corpus, and  $n^w$  is the number of documents containing the word  $w$ . As the result, the  $tf-idf$  score of the word  $w$  occurring in the document  $d$  is calculated as

$$tf-idf(w, d, D) = tf(w, d) \times idf(w, D)$$

However, when applying the tf-idf method to generate the image's textual representation, the feature vector is high-dimensional and very sparse, and its values do not reflect the semantic meaning.

Apparently, word representation acts as a basic block to build text representation and the context of words, such as co-occurrence of words in documents, affect the similarity of word meanings [93]. Instead of representing each word as the sparse and count-based vector, word embeddings [8, 25, 143] represent words as dense, low-dimensional and real-valued vectors, known as *word features*. Embedding vectors can be generated from the co-occurrence data through several reduction techniques, such as singular value decomposition (SVD) [29, 30, 125] and models generating these representations are known as *count models*. Other methods, which learn embeddings for a target word through the training

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

process in a specific corpus, are known as *predict models*. Baroni *et al.* [7] compared two types of embedding methods and proved that the predict models outperform the count ones. Thus, the predict models are candidate methods to obtain useful word representation for the proposed approaches.

### Word2Vec

One of the most popular approaches, which is inspired by neural networks, is quoted as *word2vec*. Actually, it is a group of two one-hidden neural networks: *Skip-Gram* and *CBOW* (continuous bag of words) [89, 90].

In general, their input and output data have the same dimension and are formed in one-hot vectors encoding words' indices. Word embeddings are learned during their training process to predict neighbors of a given word, including its past and future words. Different from a neural network, the non-linear/linear layer of the networks is replaced by a projection layer to obtain latent features of the input words.

While the *Skip-gram* is based on a given word to predict its context, the CBOW estimates a missing word based on its given neighboring words. The simple CBOW and Skip-gram networks taking care of one word behind and one word ahead of a given word as the context are illustrated in Figure 8.5.

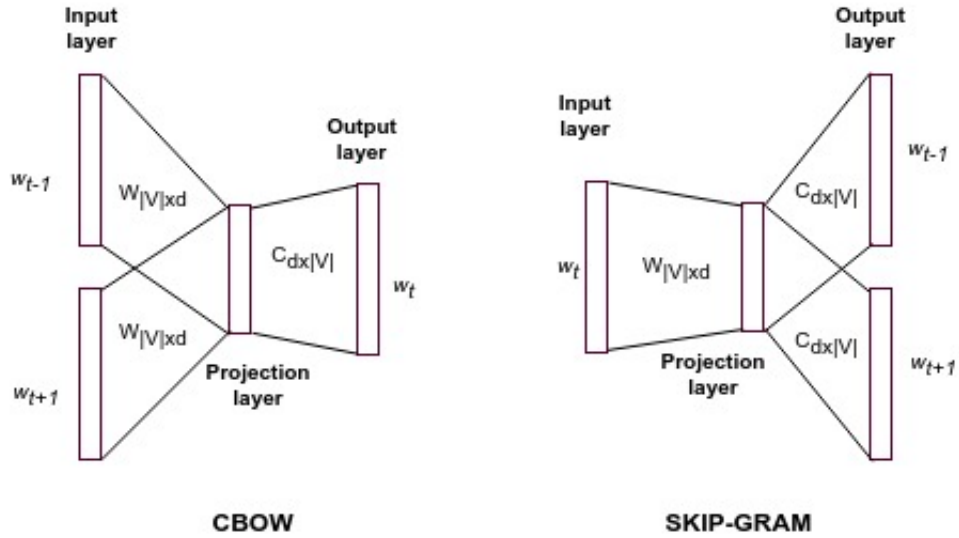


Figure 8.5: The simple architectures of the CBOW and Skip-gram network.

Let us consider a dictionary holding  $\mathcal{V}$  vocabularies. As seen in the figure, both models seek to learn two parameter matrices:  $W \in \mathbb{R}^{|\mathcal{V}| \times d}$ , in which each row is an embedding of a word in the dictionary, and  $C \in \mathbb{R}^{d \times |\mathcal{V}|}$  used to predict an expected word. The layers of those networks are fully-connected and their hidden layer has  $d$  neurons.

In the CBOW model, the neighboring words of a target word detected by a context

window having size  $L$  are fed to the network and the average values of their embeddings are used to predict the probabilities of the target word. For instance, the context window contains one word before and after a given word  $w_t$ , denoted as  $w_{t+1}$  and  $w_{t-1}$ . In addition, the indices of  $w_t$ ,  $w_{t+1}$ ,  $w_{t-1}$  in  $\mathcal{V}$  are  $i$ ,  $j$ ,  $k$  respectively. The hidden latent features of the context are the average product of the context inputs and the embedding weights as

$$\mathbf{h} = \frac{1}{2} \mathbf{W}^T (\mathbf{w}_{t-1} + \mathbf{w}_{t+1})$$

The score of each word is the product of the average embeddings and the context weights as

$$\mathbf{u} = \mathbf{C}^T \mathbf{h}$$

The probability of the target word is estimated via softmax function which generates a multinomial distribution.

$$\hat{y}_{w_t} = p(w_t | w_{t-1}, w_{t+1}) = \frac{\exp(u_i)}{\sum_{i'=1}^{|\mathcal{V}|} \exp(u_{i'})}$$

In contrast, the skip-gram predicts  $L$  context words before and after a target word so its process is like the reverse of the CBOW model. Now, the input of the network is the given center word, and the output layer is expected to predict probabilities of its neighboring words. The embedding  $\mathbf{h}_w$  of the target word  $w_t$  is the  $i$ -th row of the embedding matrix  $\mathbf{W}$  and it is computed by multiplying the one-hot vector and the parameter matrix. The context embeddings of the surroundings, named as the context vectors, are columns of  $\mathbf{C}$  at the associated positions. For examples, the context vectors of  $w_{t+1}$  and  $w_{t-1}$  are  $\mathbf{C}_{:,j}$  and  $\mathbf{C}_{:,k}$ . The probability of a context word is computed by applying the softmax operator to normalize the dot product of the target embedding and its context vector.

The goal of the training process is to maximize the probability of the actual target word given its surrounding words or the probability of the context given its center word. We refer the reader to relevant papers [39, 89, 90, 121].

#### Glove

While the architectures of the word2vec family depend on the neural networks, the heart of Glove (Global Vectors for Word Representation) [104] is word co-occurrence data within a text corpus. In the co-occurrence matrix  $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ , each cell  $X_{ij}$  indicates the frequency of a word  $w_i$  appearing in some context  $w_j$ . Similar to the matrix factorization, the high-dimensional input matrix is decomposed into a product of lower-dimensional matrices. In fact, the factorization matrix can be understood as the embedding component in which each row is the representation of a word. The embedding values are learned to minimize the difference between the co-occurrence frequencies and the products of all word pairs in the corpus.

Instead of measuring directly the error of the embedding product and the co-occurrence value, the author proposed a soft constraint for each word pair of  $i$ -th and  $j$ -th words such

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

that

$$W_i^T W_j + b_i + b_j = \log X_{ij} \quad (8.7)$$

where  $b_i$  and  $b_j$  are bias terms associated with two words. Then, the cost function which is minimized to find parameters evaluates all squared errors based on the weighted constraints

$$\mathcal{L} = \sum_{i,j=1}^{|V|} f(X_{ij})(W_i^T W_j + b_i + b_j - \log X_{ij})^2 \quad (8.8)$$

To prevent the dominance of common words during the learning process, the weighting function is defined as

$$f(X_{ij}) = \begin{cases} \left( \frac{X_{ij}}{x_{max}} \right)^\alpha & \text{if } X_{ij} < x_{max} \\ 1 & \text{otherwise} \end{cases} \quad (8.9)$$

where  $x_{max}$  and  $\alpha$  are hyper-parameters.

### FastText

Most word embedding models learn representations of complete words in the training corpus and ignore the internal structure of these words. Literally, the corpus can contain large or rare vocabularies appearing infrequently, which can be divided into sub-words found more often in the corpus. Thus, we can possibly obtain their embeddings by aggregating their sub-words' features, which are learned by some embedding extractor. FastText [14, 58], an extension of word2vec, represents a word by averaging the representations of its n-gram sub-words. For example, given a word  $w$  and  $\mathcal{G}_w$  is the set of  $n$ -gram words appearing in  $w$ . For each n-gram  $g$ , the embedding vector  $z_g$  represents the property of  $g$  and each context  $c$  of the word  $w$  has a embeddings denoted as  $v_c$ . The score of the word  $w$  given the context  $c$  is the product of the aggregation of n-grams and the representation of the context defined as

$$s(w, c) = \sum_{g \in \mathcal{G}_w} z_g^T v_c \quad (8.10)$$

### Word Embedding Models as Textual Extractors

Consider a text modeling architecture  $\mathcal{T}_{\text{glove/w2v/fastt}}$ , such as Glove, Word2Vec or FastText, in which each word is mapped to a dense  $n$ -dimensional vector. A  $j$ -th row,  $w_j$ , of the embedding matrix  $\mathcal{W} \in \mathbb{R}^{|D_S| \times n}$ , which is obtained by learning the semantic correlation of  $|D_S|$  words, represents a  $j$ -th vocabulary appearing in the source dictionary  $D_S$ . While the source dataset is a large corpus with  $|D_S|$  vocabularies, the target collection contains less unique words than the source with  $|D|$  vocabularies where  $|D| \ll |D_S|$  and

the  $D_S$  comprises nearly all of  $D$ 's vocabularies. For this reason, the source embeddings are used to provide the textual representations for all target words.

The pre-learned embeddings are used for the target task, the tag recommendation, in a static way; i.e, the embedding weights are fixed when we use them to project the representation of target words. The target embedding matrix  $\vartheta \in \mathbb{R}^{|D| \times n}$  is earned by mapping each target vocabulary with the source dictionary. If the  $k$ -th target word is found in the source dictionary associated with the representation  $\nu_j$ , the values are transferred to the  $k$ -th row of  $\vartheta$ . Otherwise, the representative values are zeros if the  $k$ -th word does not appear in the source dictionary.

To obtain the textual features of a given image, the average values of embeddings associated with tokens of the image is computed according to the frequencies and  $\vartheta$ . In the word co-occurrence perspective, each image is characterized by a high dimensional sparse vector  $r_i^{we} \in \mathbb{R}^{|D|}$ . The textual features of the image are the projection of the sparse vector into a low dimensional space according to the dictionary  $\vartheta$ .

$$z_i^{we} := f_{glove/w2v/fastt}^{we}(r_i^{we}; \vartheta) : \mathbb{R}^{|D|} \rightarrow \mathbb{R}^n$$

where

$$f_{glove/w2v/fastt}^{we}(r_i^{we}; \vartheta) = \frac{\vartheta^T r_i^{we}}{\sum_{j=1}^{|D|} (r_i^{we})_j}$$

and  $n = 300$  is the length of textual feature vector. Before employed to the tag recommendation, the word embedding models are trained in various domains to learn the *teacher's* knowledge. In particular, for the word2vec-based model, Google News dataset containing around 100 billion words is used to generate embeddings for 3 million words. For the Glove-based extractor, Common Crawl dataset consisting of 42 billion words is considered as the source knowledge of 1.9 million embeddings. FastText is also pre-learned 2 million embeddings on Common Crawl originally including 600 billion tokens [91].

### 8.3.3 Object-Detected Feature Extraction

For the comparison between different combinations of features, the representation, which is illustrated for objects captured in images, is applied in a similar way to the previous approach. As in Chapter 7, a member of YOLO family[110, 111, 112], YOLO9000, is adapted as an object-based feature extractor. While detected objects are labeled by a few common and general categories, classification datasets have a wider range of labels. By combining detection and classification dataset, the authors want to expand categories of detected objects such that the models are capable to assign objects to more particular classes, such as *watermelon* instead of *fruit*.

The authors construct a tree of visual concepts based on the WordNet, named WordTree. Instead of applying a softmax for all labels, the models perform multiple softmax operations over a group of words with the same hyponym, and the probability of a label is the multiplication of conditional probabilities with regard to the tree route associated with

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

the label. After combining data from various sources and training Darknet19 on ImageNet and COCO, YOLO9000 can detect more than 9000 object categories.

We adopt the YOLO9000 pre-trained model to study object-based features of the tag recommendation dataset. The final object-based representation of an image, which is used as an image-based factor in the proposed model, is probabilities of objects detected in the image. Furthermore, there are less categorical objects than 9000 classes revealed in the source dataset so the length of object-feature vectors is small, around less than 200 units. We denote the process as

$$z_i^{od} := f_{yolo}^{od}(R_i^{obj}) : \mathbb{R}^{448 \times 448 \times 3} \rightarrow \mathbb{R}^q \quad (8.11)$$

where  $q \ll 9000$  is the number object categories included in the tag recommendation data

### 8.3.4 Factorization Models

A factorization model based on FM or PITF model, which captures the interaction between user's/image's and tag's latent features, enables to be modified so that it will receive multiple dense and sparse values as an input of the estimator. In general, an input of the predictor is the concatenation of different features describing three dataset entities including users, images, and tags.

$$\begin{aligned} x_{u,i,t} &= \left( \underbrace{\dots, \overbrace{1}^u, \dots}_{|U|}, \underbrace{(z_i^{ic})_1, \dots, (z_i^{ic})_m}_m, \underbrace{(z_i^{we})_1, \dots, (z_i^{we})_n}_n, \underbrace{\dots, \overbrace{1}^t, \dots}_{|T|} \right) \\ &= [x_u^U, z_i^{ic}, z_i^{we}, x_t^T] \end{aligned} \quad (8.12)$$

where  $x_u^U$ ,  $x_t^T$  are one-hot vectors encoding the user's and tag's indicators.

The FM-based and PITF-based functions described in Equation 7.7 and 7.8 are employed to predict scores of tags. Depending on the types of feature's combinations including the visual-textual, visual-object and textual-object concatenations, the associated parts are replaced by the appropriate values. For example, if both visual and textual features are aggregated as the representation of an image, the FM-based function is rewritten as

$$(\hat{y}_{u,i})_t = w_t + \sum_{j=1}^p \left( V_{u,j}^U + \sum_{a=1}^m (z_i^{ic})_a \cdot V_{a,j}^{ic} + \sum_{a=1}^n (z_i^{we})_a \cdot V_{a,j}^{we} \right) V_{t,j}^T \quad (8.13)$$

where  $w_t$  is the global weight of the tag  $t$ , and  $V^U \in \mathbb{R}^{|U| \times p}$ ,  $V^{ic} \in \mathbb{R}^{m \times p}$ ,  $V^{we} \in \mathbb{R}^{n \times p}$  and  $V^T \in \mathbb{R}^{|T| \times p}$  are factorization weights. Similar to Chapter 7, the proposed models are named based on their base models, such as FM or PITF, and the image's features. Besides some models defined in the previous section, such as **FM-IC-OD** and **PITF-IC-OD**, we denote several additional models as

- Using textual features of images: **FM-WE** and **PITF-WE**
- Using the combination of visual and textual features: **FM-IC-WE**, **PITF-IC-WE**



### 8.3.5 Factorization Machine with Tag Features

Instead of representing tags by one-hot feature vectors associated with tags' indices, they can be constituted by textual features through the textual feature extraction. Let  $r_t^{tag} \in \mathbb{R}^{|D|}$  be a one-hot  $|D|$ -dimensional vector of the tag  $t$  where the element associated with the tag found in the dictionary  $D$  is 1, and 0 otherwise. If the tag is not found in the dictionary, the  $r_t^{tag}$  is a zero vector. The tag's features are obtained in a similar way as the feature extraction process by multiplying the one-hot vector and the textual weights.

$$z_t^{Twe} := f_{glove/w2v/fastt}^{we}(r_t^{tag}; \vartheta) : \mathbb{R}^{|D|} \rightarrow \mathbb{R}^n$$

where

$$f_{glove/w2v/fastt}^{we}(r_t^{tag}; \vartheta) = \vartheta^T r_t^{tag}$$

The input of the factorized predictor is extended by adding the tag's features into Equation 8.12 to produce a new input as

$$x_{u,i,t} = [x_u^U, z_i^{ic}, z_i^{we}, x_t^T, z_t^{Twe}] \quad (8.14)$$

Based on the FM-based predictor described in Equation 8.13 and the similarity between feature domains of the textual image-based and tag-based features, the interaction between images, and tags are segregated in regard to the feature domain. Namely, the visual image elements interact with the one-hot tag element while the textual features of images reach out to the textual components of tags. Thus, the score estimator is formulated as

$$\begin{aligned} (\hat{y}_{u,i})_t = & w_t + \sum_{j=1}^p \left( V_{u,j}^U + \sum_{a=1}^m (z_i^{ic})_a \cdot V_{a,j}^{ic} \right) V_{t,j}^T \\ & + (z_t^{Twe})' w^{Twe} + \sum_{j=1}^p \left( \sum_{a=1}^n (z_i^{we})_a \cdot V_{a,j}^{we} \right) \cdot \left( \sum_{b=1}^n (z_t^{Twe})_b \cdot V_{b,j}^{Twe} \right) \\ & + \sum_{a=1}^{n-1} \sum_{b=a+1}^n (z_t^{Twe})_a (z_t^{Twe})_b \sum_{j=1}^p V_{a,j}^{Twe} V_{b,j}^{Twe} \end{aligned} \quad (8.15)$$

where  $(z_t^{Twe})'$  is the transposed vector of the tag features;  $V^U$ ,  $V^{ic}$ ,  $V^{we}$  are factorized weights associated with users, visual and textual factors of images;  $V^T \in \mathbb{R}^{|T| \times p}$  and  $V^{Twe} \in \mathbb{R}^{n \times p}$  are factorization parameters belonging to the index-based and semantic-based representation of tags. The model is called **FM-IC-WE-TAG** reference for the model's name during the later evaluation section.

### 8.3.6 Personalized Neural Networks

In the approach, we again employ the personalized multi-layer perceptron as the predictor, as described in Chapter 6. The personalized layer can be applied individually to each of image features, or applied to the composite representation of images.

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

### 8.3.6.1 The One-Branch Neural Network

The proposed network with a one-branch MLP to predict scores of tags is illustrated in Figure 8.6. In this model, known as **1PerMLP**, we will utilize the modification of the

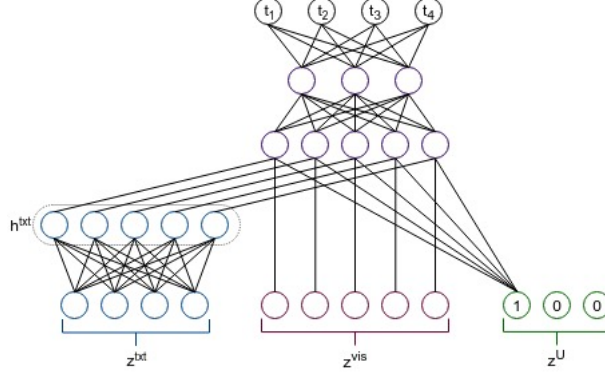


Figure 8.6: The neural network personalizes the visual-textual features to predict scores of tags.

personalized fully-connected layer described in Equation 6.10. The user-bias term is added to each element of visual features, along with adding a textual value. Before textual latent values are fed to the personalized layer to enrich the visual representation of the image, the textual input is converted to the similar space with the visual input via the projection as

$$h_i^{we} = \mathcal{H} \cdot z_i^{ic} \quad (8.16)$$

where  $\mathcal{H} \in \mathbb{R}^{m \times n}$  is the projected weights of the textual input vector. For the purpose of enhancing each element of the visual features by the textual factor during the personalizing process, Equation 6.10 is extended by the weighted textual values defined as

$$z_{u,i} = f^{per}(z_i^{ic}, h_i^{we}, x_u^U; \psi) = \sigma(b + V^U x_u^U + w^{ic} \odot z_i^{ic} + w^{we} \odot h_i^{we}) \quad (8.17)$$

where  $b \in \mathbb{R}^m$  is the bias vector;  $w^{ic} \in \mathbb{R}^m$ ,  $w^{we} \in \mathbb{R}^m$ ,  $V^U \in \mathbb{R}^{m \times |U|}$  are the weights of the visual, textual and user factors. The linear combination of various elements is followed by an elementwise non-linear function  $\sigma$  as in the fully-connected layer of a neural network. The output of the layer is a feature vector standing for the given post including the visual, textual and user information, and is fed to a one-hidden network to obtain scores of tags. The prediction function can be re-formulated as

$$\hat{y}_{u,i} = f(x_u^U, z_i^{ic}, z_i^{we}; \omega) = f^{mlp}(z_{u,i}; \xi) \quad (8.18)$$

where  $\omega = \{\mathcal{H}, \psi, \xi\}$  is the trainable parameter set of the proposed model. Scores of tags are computed in reference to the representation of a given post; i.e, parameters associated with tags are shared to all types of image's features and users as the FM-based model.

### 8.3.6.2 The Two-Branch Neural Networks

Instead of merging all features of users and images by utilizing a personalized layer, the visual and textual representations of images are fed to two branches of a neural network to predict tag scores, and the computed values of these branches are summed to obtain the final tag scores. Depending on how the user's information is employed to personalize the image's features, we construct the two-branch networks in two different ways: adding the user's representation to each element of the image's features and multiplying the network's weights by the user's weighting values.

#### User-bias as the adding element

The main idea of the proposed networks is using different user-bias variables for distinct image features and applying two non-shared MLPs to predict scores of tags in accordance with the image properties. Directly, each branch of the network gains personalized features of one given image's property by adding user-bias information to each element of the feature vector. From that idea, the architecture of the predictor, called **2PerMLP**, is modeled as in Figure 8.7. Founded on the Equation 6.10, the personalizing process for

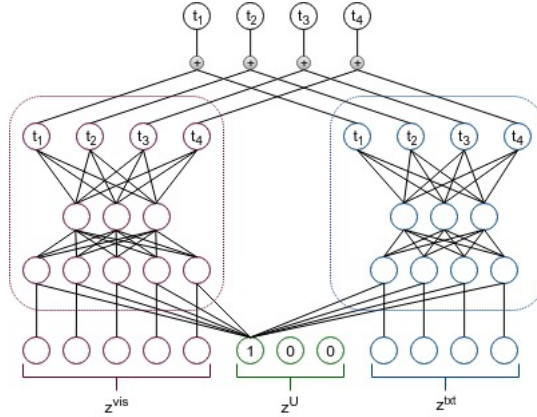


Figure 8.7: The neural network composed of two prediction branches in accordance with different types of image's features.

each element of image's feature can be defined as

$$\begin{aligned} h_{u,i}^{ic} &= f^{per}(z_i^{ic}, x_u^U; \psi^{ic}) = \sigma(b^{ic} + w^{ic} \odot z_{i,q} + V^{ic} x_u^U) \\ h_{u,i}^{we} &= f^{per}(z_i^{we}, x_u^U; \psi^{we}) = \sigma(b^{we} + w^{we} \odot z_{i,q} + V^{we} x_u^U) \end{aligned} \quad (8.19)$$

where the personalized parameters associated with visual elements  $\psi^{ic}$  contain the bias parameters  $b^{ic} \in \mathbb{R}^m$ , the weights of visual features  $w^{ic} \in \mathbb{R}^m$ , the projected parameters of the user's factors  $V^{ic} \in \mathbb{R}^{m \times |U|}$ . For the textual branch, the personalized textual-parameters  $\psi^{we}$  include  $b^{we} \in \mathbb{R}^n$ ,  $w^{we} \in \mathbb{R}^n$ , and  $V^{we} \in \mathbb{R}^{n \times |U|}$ .

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

These produced features are fed to several fully-connected layers to predict different scores of tags conjoined with the assorted combination of users and image features. The final prediction function can be seen as the aggregation of two functions associated with two branches of the network.

$$\begin{aligned}\hat{y}_{u,i} &= f(x_u^U, z_i^{ic}, z_i^{we}; \omega) = f^{ic}(x_u^U, z_i^{ic}; \omega^{ic}) + f^{we}(x_u^U, z_i^{we}; \omega^{we}) \\ &= f^{ic}(h_{u,i}^{ic}; \xi^{ic}) + f^{we}(h_{u,i}^{we}; \xi^{we})\end{aligned}\quad (8.20)$$

where  $\omega^{ic}$ ,  $\omega^{we}$  are parameters of these sub-networks correspondingly including personalized parameters and parameters related to predicted tags' scores. In the network, the factorized tag's parameters are separately defined for each of the image representations. Through the configuration, the proposed model expresses similarities to the PITF-based model so that the performances of the model is expected to be equal or possibly higher than that of the PITF-based model.

In order to take advantage of the support from other image properties, a version of the two-branch network, named as **2PerMLP-3fa**, is offered in which the personalization process is formulated as the one-branch network, described in Figure 8.8. Indeed, each

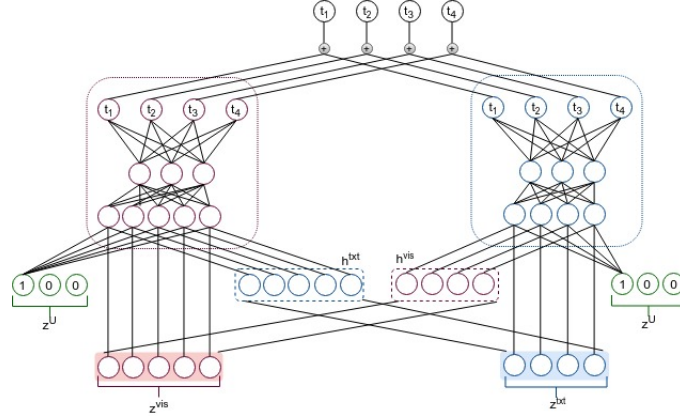


Figure 8.8: The neural network is also composed of two prediction branches in which each element of an image property interacts with other factors to obtain the new representative element.

image property, visual or textual, acts as the primary features fed to a branch and other inputs are converted to its similar space. The process is summed up as

$$h_i^{we} = \mathcal{H}^{we} \cdot z_i^{we} \quad h_i^{ic} = \mathcal{H}^{ic} \cdot z_i^{ic} \quad (8.21)$$

where  $\mathcal{H}^{we} \in \mathbb{R}^{m \times n}$ ,  $\mathcal{H}^{ic} \in \mathbb{R}^{n \times m}$  are the converted weights of the textual input and visual input.

A personalized layer, which is build based on Equation 8.17, follows the projection to capture the interaction between users and images. The output of each personalized

layer is the individual representation of each image's property under the influence of the remaining property.

$$\begin{aligned} z_{u,i}^{ic} &= f^{per}(z_i^{ic}, h_i^{we}, x_u^U; \psi^{ic}) = \sigma(b^{ic} + \mathcal{U}^{ic} \cdot x_u^U + w^{ic} \odot z_i^{ic} + v^{ic} \odot h_i^{we}) \\ z_{u,i}^{we} &= f^{per}(z_i^{we}, h_i^{ic}, x_u^U; \psi^{we}) = \sigma(b^{we} + \mathcal{U}^{we} \cdot x_u^U + w^{we} \odot z_i^{we} + v^{we} \odot h_i^{ic}) \end{aligned} \quad (8.22)$$

where the personalized parameter set of the visual branch  $\psi^{ic}$  contains the user-related variables  $\mathcal{U}^{ic} \in \mathbb{R}^{m \times |U|}$ , the visual weights  $w^{ic} \in \mathbb{R}^m$ , the textual contribution weights  $v^{ic} \in \mathbb{R}^m$ , and the bias parameters  $b^{ic} \in \mathbb{R}^m$ . Likewise, in the textual branch, parameters  $\psi^{we}$  include  $\mathcal{U}^{we} \in \mathbb{R}^{n \times |U|}$ ,  $w^{we} \in \mathbb{R}^n$ ,  $v^{we} \in \mathbb{R}^n$  and  $b^{we} \in \mathbb{R}^n$ . To compute scores of tags, these individualized features in each branch are fed to a particular MLP network and the sum of two branches' scores is the final expected output.

### User-bias as the weighting element

In preference to adding the user's information to each image features, it is possible to personalize the network parameters; i.e., each user owns a network with image data used as the input. However, it is not a good choice because of the limitation of training data for each network. We propose an approach, which personalizes synaptic weights of the input and hidden by two weighting factors, as illustrated in Figure 8.9. To cope with this, we

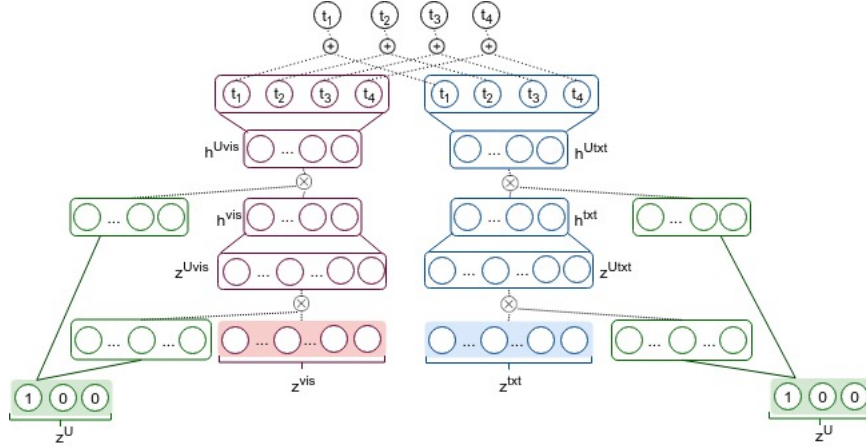


Figure 8.9: The network contains two non-shared image's information branches in which the weights of hidden layers are personalized through two element-wise steps.

weight the input and the hidden output by multiplying them with different user's values in the element-wise way. The forward process now becomes

$$h_{u,i}^* = \sigma \left( \mathcal{U}_u^* \odot \left( b^* + \mathcal{W}^* (\mathcal{V}_u^* \odot z^*) \right) \right) \quad (8.23)$$

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

where  $*$  takes the place of  $ic$  or  $we$ ;  $\mathcal{U}^{ic} \in \mathbb{R}^{|U| \times m}$ ,  $\mathcal{U}^{we} \in \mathbb{R}^{|U| \times n}$ ,  $\mathcal{V}^{|U| \times p}$  are the personal weights of the visual input, textual input, and the hidden output;  $p$  is the number of hidden neurons;  $\mathcal{W}^{ic} \in \mathbb{R}^{m \times p}$ ,  $\mathcal{W}^{we} \in \mathbb{R}^{n \times p}$  are the synaptic weights between the input and the hidden layer. Then, a fully-connected layer known as the output layer of the network receives the latent features from the hidden to estimate scores of tags.

### 8.3.7 Optimization

BPR criterion is also used to learn the model’s parameters such that the difference between scores of relevant and irrelevant tags is maximal. The gradient descent optimization algorithms with respect to quadruples  $(u, i, t^+, t^-)$  or posts  $(u, i)$  are utilized to learn parameters of the predictor. While SGD with Momentum [105] with respect to posts  $(u, i)$  is used for the MLP-based models, the Adagrad [33] algorithm, which adapts the learning rate to the parameters, is utilized to find parameters of the factorization-based models.

In SGD with momentum optimization as described in Algorithm 4, for each post selected randomly from the training post set, the sparse  $|D|$ -dimensional vector is created in relation to the frequency mapping of the given image  $R_i^{txt}$ . In addition, we sample a subset  $\tilde{T}_{u,i}^-$  of irrelevant tags connected with the given post. Visual features  $z_i^{ic}$  are extracted based on the color image  $R_i^{vis}$  and the fixed visual extractor  $f^{ic}$ , while textual features  $z_i^{txt}$  is obtained by the extracting function  $f^{we}$  receiving the token-frequency vector  $r_i^{we}$  as its input. The loss value  $\text{BPR}_{u,i}$  of the post is computed based on the relevant set  $T_{u,i}^+$  and irrelevant subset  $\tilde{T}_{u,i}^-$ , as in Equation 5.17. The derivative of  $\text{BPR}_{u,i}$  with respect to the predictor’s parameters  $\omega$  is found to update the momentum and the parameters.

From another point of view, in Algorithm 5, a random triple  $(u, i, t^+)$  is selected from the training assignment set  $\mathcal{S}_{train}$ . Then, an irrelevant tag is arbitrarily drawn out of the irrelevant tag set of the post. The discrepancy of two tags’ scores described in Equation 7.12 is computed such that the predictor’s parameters are updated by the gradients of  $\text{BPR}_{u,i,t^+,t^-}$  with respect to these variables. Moreover, the learning rate is adopted in accordance with these gradients. In both algorithms, parameters of the extractors are fixed with the knowledge received from the source models so the optimization process solely seeks to learn parameters of the predictor.

## 8.4 Evaluation

### 8.4.1 Dataset

We obtained experiments on NUS-WIDE-3 described in Chapter 4. Similar to Chapter 7, color photos were scaled into  $224 \times 224$  pixels used to extract visual features and  $448 \times 448$  pixels for the object-detected feature extraction. After cleaning text, nearly 97 percent of images contain text used to obtain textual features. Totally, more than 95 percent of images are tagged by one user, and there are averagely six tags assigned to an image by a specific user.

**Algorithm 4** Learning BPR with respect to posts

---

```

1: Input:  $P_S, \mathcal{R}^{vis}, \mathcal{R}^{txt}, \theta, \vartheta, \alpha, \mu, N$ 
2: Output:  $\omega$ 

3: Initialize  $\omega \leftarrow \mathcal{N}(0, 0.1)$ 
4:  $\mathcal{M} \leftarrow 0$ 
5: repeat
6:   Pick randomly  $(u, i) \in P_{S_{train}}$ 
7:   Get RGB data  $R_i^{vis} \in \mathcal{R}^{vis}$ , token's frequencies  $R_i^{txt} \in \mathcal{R}^{txt}$  of the image  $i$ 
8:    $r_i^{we} \leftarrow \text{convert\_to\_vector}(R_i^{txt})$ 
9:   Sample  $\tilde{T}_{u,i}^- \subset T_{u,i}^-$ , where  $|\tilde{T}_{u,i}^-| = N$ 
10:   $z_i^{ic} \leftarrow f^{ic}(R_i^{ic}; \theta)$ 
11:   $z_i^{we} \leftarrow f^{we}(r_i^{we}; \vartheta)$ 
12:   $\hat{y}_{u,i} \leftarrow f(x_u, z_i^{we}, z_i^{ic}; \omega)$ 
13:   $\text{BPR}_{u,i} \leftarrow 0$ 
14:  for  $t^+ \in T_{u,i}^+$  do
15:    for  $t^- \in \tilde{T}_{u,i}^-$  do
16:       $\text{BPR}_{u,i} \leftarrow \text{BPR}_{u,i} + \ln \sigma \left( (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-} \right)$ 
17:    end for
18:  end for
19:   $\text{BPR}_{u,i} \leftarrow \frac{\text{BPR}_{u,i}}{|T_{u,i}^+| \cdot |\tilde{T}_{u,i}^-|}$ 
20:   $g \leftarrow \alpha \left( \frac{\partial \text{BPR}_{u,i}}{\partial \omega} \right) + \mu \cdot \mathcal{M}$ 
21:  Update  $\omega \leftarrow \omega + g$ 
22:  Update  $\mathcal{M} \leftarrow g$ 
23: until convergence
24: return  $\omega$ 

```

---

**8.4.2 Experimental Setup**

The VGG-based extractors including VGG-16 and VGG-19 provide 512-dimensional visual feature vectors while the extractor relying on the ResNet50 network generates 2048-dimensional vectors representing images. Furthermore, the textual features extracted by the word-embedding models are included in 300-dimensional vectors. For comparison purposes, we work with object-detected features comprised in 186-dimensional vectors involving probabilities of objects appearing in images. The factor dimension for factorization architectures and the size of hidden layers are fixed in 128.

The evaluation metric used in this paper is the F1-measure as described in Equation 5.24. For factorization predictors, the learning rate  $\alpha$  were searched within the range  $\{0.1, 0.01, 0.001\}$ , and the search range is  $\{0.01, 0.001\}$  for MLP predictors. In both types

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---



---

### Algorithm 5 Learning BPR with respect to triples

---

```

1: Input:  $S, \mathcal{R}^{vis}, \mathcal{R}^{txt}, \theta, \vartheta, \alpha, \epsilon, N$ 
2: Output:  $\omega$ 

3: Initialize  $\omega \leftarrow \mathcal{N}(0, 0.1)$ 
4:  $\mathcal{G} \leftarrow 0$ 
5: repeat
6:   Pick randomly  $(u, i, t^+) \in S_{train}$ 
7:   Get RGB data  $R_i^{vis} \in \mathcal{R}^{vis}$ , token's frequencies  $R_i^{txt} \in \mathcal{R}^{txt}$  of the image  $i$ 
8:    $r_i^{we} \leftarrow \text{convert\_to\_vector}(R_i^{txt})$ 
9:   Sample  $t^- \in T_{u,i}^-$ 
10:   $z_i^{ic} \leftarrow f^{ic}(R_i^{ic}; \theta)$ 
11:   $z_i^{we} \leftarrow f^{we}(r_i^{we}; \vartheta)$ 
12:  Compute  $(\hat{y}_{u,i})_{t^+}$  and  $(\hat{y}_{u,i})_{t^-}$ 
13:   $\text{BPR}_{u,i,t^+,t^-} \leftarrow \ln \sigma \left( (\hat{y}_{u,i})_{t^+} - (\hat{y}_{u,i})_{t^-} \right)$ 
14:   $\Delta \leftarrow \frac{\partial \text{BPR}_{u,i,t^+,t^-}}{\partial \omega}$ 
15:  Update  $\mathcal{G} \leftarrow \mathcal{G} + \Delta^2$ 
16:  Update  $\omega \leftarrow \omega + \frac{\alpha}{\sqrt{\mathcal{G} + \epsilon}} \cdot \Delta$ 
17: until convergence
18: return  $\omega$ 

```

---

of predictor architectures, the L2-regularization is used to avoid the over-fitting with the searching range  $\{1e-4, 1e-5, 1e-6, 1e-7\}$ . The momentum  $\mu$  is also fixed to 0.9 during the training phase.

The proposed models are compared to following personalized tag recommendation approaches that are based only on the users' preference: **PITF** and **FM**. For the comparison between factorization and neural network architectures for tag recommendation, the VGG-16 and Glove features are used. These features are also applied for the comparison among different representations of images. For evaluating the effectiveness of transferred knowledge, the factorization models using different textual and visual features are conducted. They include

- Models with on visual features: **FM-VGG16**, **FM-VGG19**, **FM-ResNet50**, **PITF-VGG16**, **PITF-VGG19**, **PITF-ResNet50**.
- Models with textual features: **FM-Glove**, **FM-W2V**, **FM-FastText**, **PITF-Glove**, **PITF-W2V**, **FM-FastText**.

### 8.4.3 Results

Besides the finding results about the effect of object-detected features on tag recommendation performance as in Chapter 7, we analyze the influence of the textual factor to the



performance of predictors when it is concatenated with the visual factor. Figure 8.10 and 8.11 describe the effect of different features of images to tag recommendation. In these figures, the **FM-OD** and **PITF-OD**, which take into account the object-based features of images, have the least improvements compared to the others. Moreover, when compared to the **FM-WE** and **PITF-WE** which consider to the textual representation of images as a part of their inputs, the visual-based tag recommendation **FM-IC** and **PITF-IC** achieve better accuracy. The improvements of the recommendation performance may implicitly mean that visual representations are more effective elements to recommend relevant tags to posts, and it is promising that the combination between the features and the textual features standing in the second place are possible to increase the accuracy of the recommender.

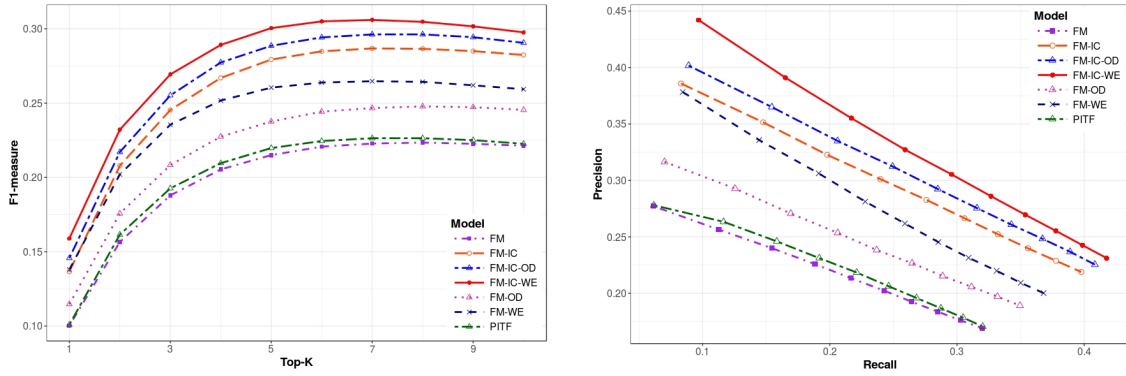


Figure 8.10: F1-measure and Precision-Recall for FM models rely on VGG-16, Glove, and YOLO extractors.

Indeed, the combination of visual and textual features applied in **FM-IC-WE** and **PITF-IC-WE** promote the better performance on recommending tags for images compared to the **FM-IC-OD** and **PITF-IC-OD** which exploit the merger of visual and object-detected features to predict scores of tags as in Chapter 7. This trend may illustrate that richer features provide the better performance.

To examine how the extractors and transferred knowledge affect the performance of the tag recommendation, we evaluate the accuracy of different factorization-based models, which receive various visual and textual features as their inputs. The results are plotted in Figure 8.12 and 8.13. We can clearly see that the visual representation is still the most effective factor to estimate personalized tags.

Among three types of the visual extractor, the ResNet network, which has achieved better performance than VGG models in image classification, produces richer feature vectors with higher length. These features increase the truthfulness of recommended tags around one percent compared to VGG-based features. Two dictionaries obtained by two textual extractors, Glove and FastText, in the similar domain Common Crawl generate lightly more effective features compared to the word2vec extractor getting knowledge in GoogleNews domain.

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

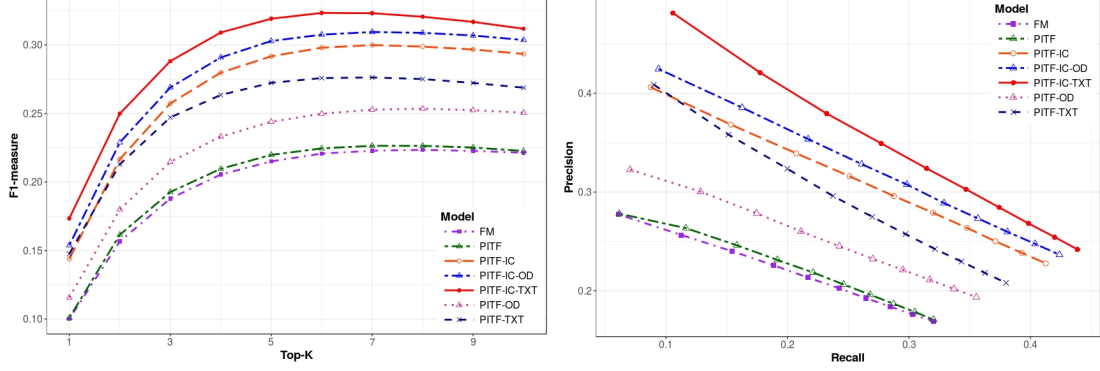


Figure 8.11: F1-measure and Precision-Recall for PITF models rely on VGG-16, Glove, and YOLO extractors.

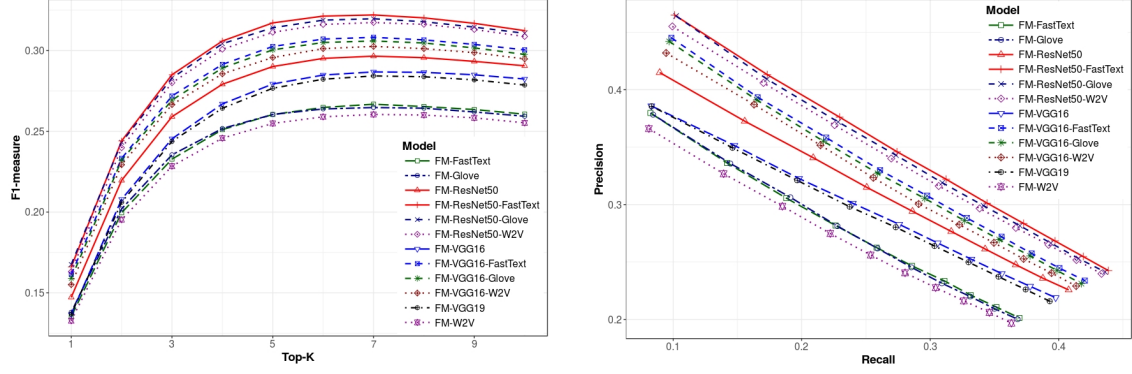


Figure 8.12: Visual and textual features extracted by different extractors boost the performance of FM-based models in different levels.

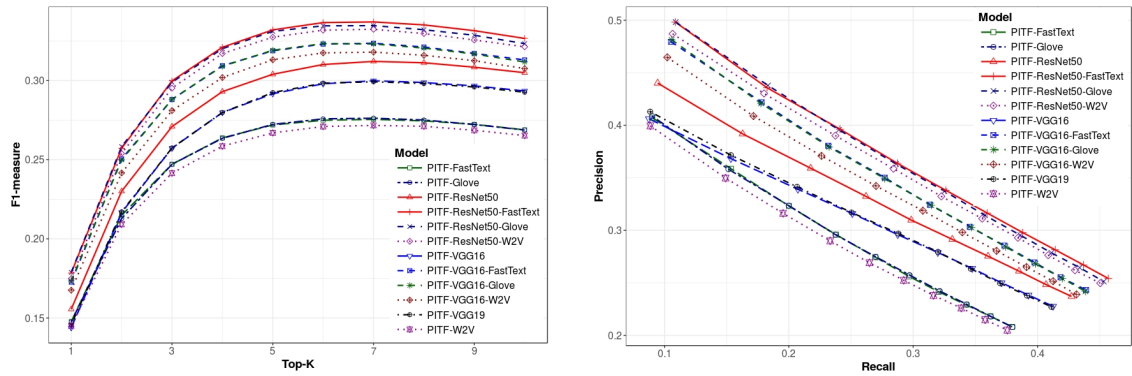


Figure 8.13: Visual and textual features extracted by different extractors boost the performance of PITF-based models in different levels.

Since the combination of different features can improve the performance of predictors, we also report the results of using different mergers as the image representation. As in the stand-alone cases, ResNet features are the best compared to other visual ones, and Glove- and FastText-features are better than word2vec textual features. Apparently, the combinations of these features, ResNet-FastText and ResNet-Glove, boost the predictors to the top of the best performance models. Although the effectiveness of using ResNet-FastText in the tag recommendation is comparatively higher than the ResNet-Glove, the contribution of the FastText and Glove is not different in the combination with the VGG-16 features.

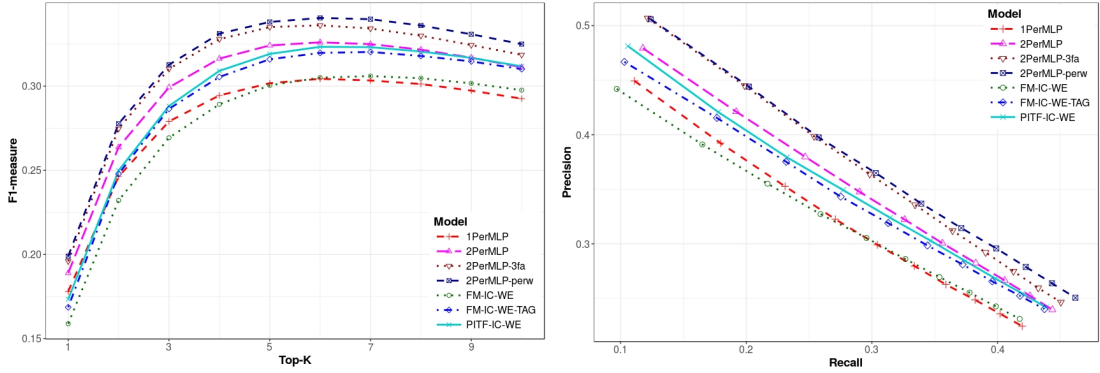


Figure 8.14: The comparison among factor-based and MLP-based models using the combination of VGG-16 and Glove features.

Figure 8.14 shows the comparison of different proposed models, factor-based and MLP-based predictors, using the combination of VGG-16 and Glove features. According to experiments in Chapter 6, the personalized MLP predictors perform more desirable than the factorization-based predictors from top-1 to top-4 accuracies, but their prediction is less accurate than the others regards to top-6 to top-10. The discussion is again reflected in the results of the personalized MLP-based models based on two types of image’s features, especially in the model **1PerMLP**.

Since the **1PerMLP** shares the tag-related parameters during capturing the interaction between tags and other factors, the model is quite similar to the **FM**-based predictor. As a result, its performance is closed to the **FM-IC-WE** using visual and textual features as the inputs on average. On the other hand, the **2PerMLP**, which separates tag parameters according to the interacted factors, is established in a partly similar way of the **FM-IC-WE-TAG** and **PITF-IC-WE**. For this reason, its accuracies are better than 1PerMLP and nearly comparable to **FM-IC-WE-TAG** and **PITF-IC-WE**.

As each personalized attribute of a given image is embellished resulting from the support of the remaining non-personalized attribute, the process of predicting tag scores is enhanced shown through the performance of **2PerMLP-3fa**. The model achieves better accuracy compared to the **2PerMLP** and **PITF-IC-WE**, which treat different image’s attributes independently during the prediction process. Additionally, the network with

## 8. PERSONALIZED TAG RECOMMENDATION USING TEXT TRANSFER LEARNING

---

the hidden parameters weighted by the user’s information **2PerMLP-perw** reaches the comparable accuracy with the above network.

### 8.5 Discussion

In this chapter, we have proposed approaches which are built based on factorization frameworks or personalized neural networks receiving the visual-textual features as their inputs to exploit the relationships among users, images, and tags in predicting scores of tags. Furthermore, we have explored the efficiency of numerous features to the performance of predictors. Experimental results show that image’s features enriched by the text data perform more excellently than the other state-of-the-art methods, which are purely based on the assignment information.

Although the proposed methods can improve the prediction results, there are still available rooms for further improvements. For instance, the text data source, which provides textual features, can be expanded by adding image’s comments. In addition, the user’s information used in the models is still the index data so the meta-data of users can be investigated to enrich the post’s features

## Chapter 9

# Conclusion

### Contents

---

<b>9.1 Thesis Conclusion . . . . .</b>	<b>105</b>
9.1.1 Content-aware Factorization Model . . . . .	105
9.1.2 Personalized Neural Network . . . . .	106
9.1.3 Object-based Features for Tag Recommendation . . . . .	106
9.1.4 Text-based Features for Tag Recommendation . . . . .	106
<b>9.2 Future Research Direction . . . . .</b>	<b>107</b>

---

This chapter concludes the thesis and summarizes our work on personalized tag recommendation for images. Future works in form of practical applications and research direction are also pointed out.

### 9.1 Thesis Conclusion

This thesis addressed the problem of personalized tag recommendation for images by following the deep learning based paradigm and leveraging historical tagging information. It discussed the importance of image-based content and context in the annotation process and presented solutions for boosting the performance and the quality of the predicted tags. More specifically, the thesis proposed a complete framework for personalized image tag recommendation which is built on the efficiency of different considerations.

#### 9.1.1 Content-aware Factorization Model

In Chapter 5, we described the improvement of a factorization based model for personalized tag recommendation. Originally, the factorization machine examines the ternary relations among users, images and tags to compute scores of tags. New images, which are not observed during the training process, are the limitation of the factorization models. To tackle the problem, we divided the tag predicting workflow into two sequential phases. A

## 9. CONCLUSION

---

feature extraction phase, which is concerned with issues related to extracting visual representations of images, whereas a computation phase, which describes the actual prediction process. In fact, the CNN with three trainable layers is obtained to generate high-level features reflecting the content of images. The generated features are fed to a FM such that the actual interaction considered in the model is among users, visual representation and tags. By using the visual factor, which is always available for any image, the cold-start item issue is solved and the predictive power of a latent factor model is maximized. Our empirical study on NUS-WIDE dataset shows that the learned features of images obtained by a CNN strongly support boosting up the performance of tag recommendation, and the model combining both the visual factor and historical tagging behavior outperforms a number of previous attempts developed in the past literature.

### 9.1.2 Personalized Neural Network

Although a pure neural network based approach is a sufficient candidate for non-personalized scenarios where all related images receive the same recommendation, it cannot capture the relation between users and tags as a factorization based model did. We have proposed the extension of a convolutional network for the personalized scenario. In our work presented in Chapter 6, to personalize learned visual features, a specific layer, which adds the user's information to each visual element, follows the extractor and its output is fed to the two-layer neural network to estimate scores of tags. We have also adapted the BPR criterion and the positive-negative sampling algorithm for post based perspective. We evaluated the different performance of various personalization levels; e.g, the first level is when the personalized layer goes after the first convolutional layer.

### 9.1.3 Object-based Features for Tag Recommendation

In Chapter 7, the thesis presented an approach for boosting the performance of content-aware models based on deep learning and factorization. Aside from the visual content, the object-based context explored from the inner side of an image is exploited to increase the recommendation performance. We believe that the proposed approaches can be further improved by considering and combining additional features for image representation. In addition to the FM-based model, the thesis presented a PITF-based method that also enables to capture the interaction among users, tags and the dense representation of images. To reduce the computation required by each extractor and deal with the lack of training data for deep learning, we adopted the transfer learning technique to pass the rich knowledge source domains to the tag recommendation domain.

### 9.1.4 Text-based Features for Tag Recommendation

Chapter 8 discussed another context-based representation which is derived from surrounding words, such as from the image's description and title. We have extended various factorization-based models that allow the image's content and context passing to compute

scores of tags. Additionally, we proposed several neural network based architectures to combine different factors of tag recommendation including users, image-based content, and context to estimate scores of tags. Through the experiments, we demonstrate the efficiency of the contextual entity to enhance the accuracy of the predictor. We also discuss how various frameworks perform when dealing with different transferred knowledge. We analyzed the connection between factorization- and neural network-based models to predict tags associated with the user's preference, visual content and semantic context.

## 9.2 Future Research Direction

We see the following general future research directions that are promising:

- **User-Based Feature Extraction.**

The results of the experiments show that texts surrounding images are truly a helpful source to provide effective features to enhance the performance of tag recommendation. The kind of features is possibly adapted to represent for a given user, who owns his dictionary created during his interaction with the system. His vocabularies can come from his comments, or image's description, and they highly correlate with annotation keywords that he probably uses to annotate images. Moreover, the context of a user, such as his location, contact, groups, is considered to represent his characteristics. Since the proposed models are flexible to change or extend the input data, they enable to consume a variety of the user's representations replacing for the index-based features.

- **Tag-based Embedding.**

As in [147], a hybrid of a CNN and a recurrent network is constructed to capture the relation between visual and semantic features during the predicting process. Besides, we also evaluated the additional features of tags enhancing the performance of the factorization based model. However, the experiments mostly focused on the interaction between two features belonging to the similar domain. Investigating to examine the relationship between visual features of images and textual features of tags/users is also another important direction to understand social tagging tasks.

- **Scalable and robust methods for tag recommendation using diverse sources.**

In this thesis, we only conducted experiments on image data sources, mainly Flickr data. However, since social media exists in a wide variety of formation, further implementation can focus on some other potential data type, such as semantic Web data or audio data. For example, for a semantic Web, the tag recommendation is constructed based on the semantic features, and mostly ignores the visual contents indicated by a few included images. However, the image-based content is possibly considered as the context factor in a tag recommendation task.

## 9. CONCLUSION

---

Furthermore, CNNs achieve not only the success in the image-related task but also prove their power in audio scenarios, such as speech recognition [2, 3]. For this reason, another direction is to extend the same approach used in this thesis to predict tags for audio resources.

- **Content-aware Recommender Systems.**

Cheng *et al.* [21] proposed a deep learning approach for recommender systems, which concatenate different features and use the combination to predict the relevant scores of items. It motivates us to investigate the proposed approaches into item recommendation scenarios, such as movie recommendation. For example, each movie is described by a short trailer or poster with its description. From these sources, the movie's representation is generated by using a CNN-based extractor and is combined with the user's information to estimate its relevance.



# References

- [1] ABBASI, R., GRZEGORZEK, M. & STAAB, S. (2009). Large scale tag recommendation using different image representations. In *International Conference on Semantic and Digital Media Technologies*, 65–76, Springer. 35
- [2] ABDEL-HAMID, O., MOHAMED, A.R., JIANG, H. & PENN, G. (2012). Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 4277–4280, IEEE. 108
- [3] ABDEL-HAMID, O., DENG, L. & YU, D. (2013). Exploring convolutional neural network structures and optimization techniques for speech recognition. In *Inter-speech*, vol. 2013, 1173–5. 108
- [4] AMES, M. & NAAMAN, M. (2007). Why we tag: motivations for annotation in mobile and online media. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, 971–980. 3
- [5] AMLACHER, K., FRITZ, G., LULEY, P., ALMER, A. & PALETTA, L. (2009). Geo-contextual priors for attentive urban object recognition. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 1214–1219, IEEE. 65
- [6] ANDERSON, A., RANGHUNATHAN, K. & VOGEL, A. (2008). Tagez: Flickr tag recommendation. *Association for the Advancement of Artificial Intelligence*. 3
- [7] BARONI, M., DINU, G. & KRUSZEWSKI, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 238–247. 88
- [8] BENGIO, Y., DUCHARME, R., VINCENT, P. & JAUVIN, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, **3**, 1137–1155. 87
- [9] BENZ, D., HOTH, A., JSCHKE, R., KRAUSE, B., MITZLAFF, F., SCHMITZ, C. & STUMME, G. (2010). The social bookmark and publication management system BibSonomy. *The VLDB Journal*, **19**, 849–875. 15

## REFERENCES

---

- [10] BERG, T., LIU, J., LEE, S.W., ALEXANDER, M.L., JACOBS, D.W. & BELHUMEUR, P.N. (2014). Birdsnap: Large-scale fine-grained visual categorization of birds. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 2019–2026, IEEE. 65
- [11] BERG, T.L. & FORSYTH, D.A. (2006). Animals on the web. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 1463–1470, IEEE. 65
- [12] BEST, P., MANKTELOW, R. & TAYLOR, B. (2014). Online communication, social media and adolescent wellbeing: A systematic narrative review. *Children and Youth Services Review*, **41**, 27–36. 1
- [13] BISHOP, C.M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. 9
- [14] BOJANOWSKI, P., GRAVE, E., JOULIN, A. & MIKOLOV, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*. 90
- [15] BOTT, E. & SPILLIUS, E.B. (2014). *Family and social network: Roles, norms and external relationships in ordinary urban families*. Routledge. 1
- [16] BROOKS, C.H. & MONTANEZ, N. (2006). Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proceedings of the 15th international conference on World Wide Web*, 625–632, ACM. 3
- [17] BUDUMA, N. & LOCASCIO, N. (2017). *Fundamentals of deep learning: designing next-generation machine intelligence algorithms*. " O'Reilly Media, Inc.". 9
- [18] CHEN, H.M., CHANG, M.H., CHANG, P.C., TIEN, M.C., HSU, W.H. & WU, J.L. (2008). Sheepdog: group and tag recommendation for flickr photos by automatic search-based learning. In *Proceedings of the 16th ACM international conference on Multimedia*, 737–740, ACM. 35
- [19] CHEN, Q., SONG, Z., DONG, J., HUANG, Z., HUA, Y. & YAN, S. (2015). Contextualizing object detection and classification. *IEEE transactions on pattern analysis and machine intelligence*, **37**, 13–27. 65, 66
- [20] CHEN, Z., CAO, J., SONG, Y., GUO, J., ZHANG, Y. & LI, J. (2010). Context-oriented web video tag recommendation. In *Proceedings of the 19th international conference on World wide web*, 1079–1080, ACM. 4
- [21] CHENG, H.T., KOC, L., HARMSSEN, J., SHAKED, T., CHANDRA, T., ARADHYE, H., ANDERSON, G., CORRADO, G., CHAI, W., ISPIR, M. *et al.* (2016). Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 7–10, ACM. 108

- 
- [22] CHUA, T.S., TANG, J., HONG, R., LI, H., LUO, Z. & ZHENG, Y. (2009). Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*, 48. 26
  - [23] CHUNG, N., LEE, S. & HAN, H. (2015). Understanding communication types on travel information sharing in social media: A transactive memory systems perspective. *Telematics and Informatics*, **32**, 564–575. 1
  - [24] CLEVERT, D.A., UNTERTHINER, T. & HOCHREITER, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*. 11
  - [25] COLLOBERT, R. & WESTON, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, 160–167, ACM. 87
  - [26] CORTES, C. & MOHRI, M. (2004). Auc optimization vs. error rate minimization. In *Advances in neural information processing systems*, 313–320. 21
  - [27] DE LATHAUWER, L., DE MOOR, B. & VANDEWALLE, J. (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, **21**, 1253–1278. 16
  - [28] DE LATHAUWER, L., DE MOOR, B. & VANDEWALLE, J. (2000). On the best rank-1 and rank-( $r_1, r_2, \dots, r_n$ ) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, **21**, 1324–1342. 16
  - [29] DEERWESTER, S., DUMAIS, S.T., FURNAS, G.W., LANDAUER, T.K. & HARSHMAN, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, **41**, 391. 87
  - [30] DEERWESTER, S.C., DUMAIS, S.T., FURNAS, G.W., HARSHMAN, R.A., LANDAUER, T.K., LOCHBAUM, K.E. & STREETER, L.A. (1989). Computer information retrieval using latent semantic structure. US Patent 4,839,853. 87
  - [31] DENG, J., DONG, W., SOCHER, R., LI, L.J., LI, K. & FEI-FEI, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255, IEEE. 29
  - [32] DIVVALA, S.K., HOIEM, D., HAYS, J.H., EFROS, A.A. & HEBERT, M. (2009). An empirical study of context in object detection. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 1271–1278, IEEE. 65
  - [33] DUCHI, J., HAZAN, E. & SINGER, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, **12**, 2121–2159. 98

## REFERENCES

---

- [34] GANTNER, Z., RENDLE, S. & SCHMIDT-THIEME, L. (2010). Factorization models for context-/time-aware movie recommendations. In *Proceedings of the Workshop on Context-Aware Movie Recommendation*, 14–19, ACM. 65
- [35] GARG, N. & WEBER, I. (2008). Personalized, interactive tag recommendation for flickr. In *Proceedings of the 2008 ACM conference on Recommender systems*, 67–74. 35, 82
- [36] GIRSHICK, R. (2015). Fast r-cnn. *arXiv preprint arXiv:1504.08083*. 65, 69
- [37] GIRSHICK, R., DONAHUE, J., DARRELL, T. & MALIK, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587. 65, 68, 69
- [38] GIRSHICK, R., DONAHUE, J., DARRELL, T. & MALIK, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, **38**, 142–158. 65
- [39] GOLDBERG, Y. & LEVY, O. (2014). word2vec explained: Deriving mikolov et al.’s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*. 89
- [40] GOLDER, S.A. & HUBERMAN, B.A. (2006). Usage patterns of collaborative tagging systems. *Journal of information science*, **32**, 198–208. 3
- [41] GONG, Y., JIA, Y., LEUNG, T., TOSHEV, A. & IOFFE, S. (2013). Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*. 36, 50
- [42] GOODFELLOW, I., BENGIO, Y. & COURVILLE, A. (2016). *Deep Learning*. MIT Press, <http://www.deeplearningbook.org>. 10
- [43] GRAHAM, B. (2014). Fractional max-pooling. *arXiv preprint arXiv:1412.6071*. 5
- [44] GUILLAUMIN, M., VERBEEK, J. & SCHMID, C. (2010). Multimodal semi-supervised learning for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 902–909, IEEE. 65
- [45] GUPTA, M., LI, R., YIN, Z. & HAN, J. (2010). Survey on social tagging techniques. *ACM Sigkdd Explorations Newsletter*, **12**, 58–72. 3
- [46] HARPER, F.M. & KONSTAN, J.A. (2015). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, **5**, 19:1–19:19. 15
- [47] HARSHMAN, R.A. (1970). Foundations of the parafac procedure: Models and conditions for an” explanatory” multimodal factor analysis. 16

- 
- [48] HARZALLAH, H., JURIE, F. & SCHMID, C. (2009). Combining efficient object localization and image classification. In *Computer Vision, 2009 IEEE 12th International Conference on*, 237–244, IEEE. 65, 66
- [49] HE, K., ZHANG, X., REN, S. & SUN, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, 346–361, Springer. 68
- [50] HE, K., ZHANG, X., REN, S. & SUN, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778. 13, 52, 85
- [51] HECKNER, M., HEILEMANN, M. & WOLFF, C. (2009). Personal information management vs. resource sharing: Towards a model of information behaviour in social tagging systems. 3
- [52] HEITZ, G. & KOLLER, D. (2008). Learning spatial context: Using stuff to find things. In *European conference on computer vision*, 30–43, Springer. 65
- [53] HOCHREITER, S. & SCHMIDHUBER, J. (1997). Long short-term memory. *Neural computation*, **9**, 1735–1780. 82
- [54] HOIEM, D., EFROS, A.A. & HEBERT, M. (2005). Geometric context from a single image. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1, 654–661, IEEE. 65
- [55] HOTH, A., JÄSCHKE, R., SCHMITZ, C., STUMME, G. & ALTHOFF, K.D. (2006). FolkRank: A ranking algorithm for folksonomies. In *LWA*, vol. 1, 111–114. 45
- [56] JÄSCHKE, R., MARINHO, L., HOTH, A., SCHMIDT-THIEME, L. & STUMME, G. (2007). Tag recommendations in folksonomies. In *Knowledge Discovery in Databases: PKDD 2007*, 506–514. 31, 45, 58
- [57] JÄSCHKE, R., MARINHO, L., HOTH, A., SCHMIDT-THIEME, L. & STUMME, G. (2008). Tag recommendations in social bookmarking systems. *Ai Communications*, **21**, 231–247. 31
- [58] JOULIN, A., GRAVE, E., BOJANOWSKI, P. & MIKOLOV, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*. 90
- [59] KAPLAN, A.M. & HAENLEIN, M. (2010). Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, **53**, 59–68. 1
- [60] KAVANAUGH, A.L., FOX, E.A., SHEETZ, S.D., YANG, S., LI, L.T., SHOEMAKER, D.J., NATSEV, A. & XIE, L. (2012). Social media use by government: From the routine to the critical. *Government Information Quarterly*, **29**, 480–491. 1

## REFERENCES

---

- [61] KIETZMANN, J.H., HERMKENS, K., MCCARTHY, I.P. & SILVESTRE, B.S. (2011). Social media? get serious! understanding the functional building blocks of social media. *Business horizons*, **54**, 241–251. 1
- [62] KIPP, M.E. & CAMPBELL, D.G. (2006). Patterns and inconsistencies in collaborative tagging systems: An examination of tagging practices. *Proceedings of the Association for Information Science and Technology*, **43**, 1–18. 3
- [63] KOLDA, T.G. & BADER, B.W. (2009). Tensor decompositions and applications. *SIAM review*, **51**, 455–500. 16
- [64] KOLDA, T.G. & SUN, J. (2008). Scalable tensor decompositions for multi-aspect data mining. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 363–372, IEEE. 16
- [65] KOREN, Y., BELL, R. & VOLINSKY, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, **42**. 15
- [66] KÖRNER, C. (2009). Understanding the motivation behind tagging. *ACM Student Research Competition-Hypertext*, **9**. 3
- [67] KOWALD, D., LACIC, E. & TRATTNER, C. (2014). Tagrec: Towards a standardized tag recommender benchmarking framework. In *Proceedings of the 25th ACM conference on Hypertext and social media*, 305–307. 45, 58
- [68] KRIZHEVSKY, A., SUTSKEVER, I. & HINTON, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105. 12, 13, 38, 52, 64, 65
- [69] LECUN, Y., BOSER, B., DENKER, J.S., HENDERSON, D., HOWARD, R.E., HUBBARD, W. & JACKEL, L.D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, **1**, 541–551. 11
- [70] LECUN, Y., BOTTOU, L., BENGIO, Y. & HAFFNER, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, **86**, 2278–2324. 12
- [71] LECUN, Y. *et al.* (1989). Generalization and network design strategies. *Connectionism in perspective*, 143–155. 12
- [72] LEE, D.D. & SEUNG, H.S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, 556–562. 16
- [73] LEE, S., DE NEVE, W., PLATANIOTIS, K.N. & RO, Y.M. (2010). Map-based image tag recommendation using a visual folksonomy. *Pattern Recognition Letters*, **31**, 976–982. 35

- 
- [74] LERMAN, K. & JONES, L. (2006). Social browsing on flickr. *arXiv preprint cs/0612047*. 26
- [75] LI, J. & WANG, J.Z. (2008). Real-time computerized annotation of pictures. *IEEE transactions on pattern analysis and machine intelligence*, **30**, 985–1002. 3
- [76] LI, L., LEBANON, G. & PARK, H. (2012). Fast bregman divergence nmf using taylor expansion and coordinate descent. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 307–315, ACM. 16
- [77] LI, X., SNOEK, C.G. & WORRING, M. (2008). Learning tag relevance by neighbor voting for social image retrieval. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, 180–187. 35
- [78] LIN, M., CHEN, Q. & YAN, S. (2013). Network in network. In *International Conference on Learning Representations*. 13, 38, 65
- [79] LIN, T.Y., MAIRE, M., BELONGIE, S., HAYS, J., PERONA, P., RAMANAN, D., DOLLÁR, P. & ZITNICK, C.L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755, Springer. 30
- [80] LIU, J., LI, Z., TANG, J., JIANG, Y. & LU, H. (2014). Personalized geo-specific tag recommendation for photos on social websites. *IEEE Transactions on Multimedia*, **16**, 588–600. 82
- [81] LIU, W., ANGUELOV, D., ERHAN, D., SZEGEDY, C., REED, S., FU, C.Y. & BERG, A.C. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37, Springer. 68
- [82] LONG, J., SHELHAMER, E. & DARRELL, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3431–3440. 68
- [83] MAAS, A.L., HANNUN, A.Y. & NG, A.Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, vol. 30, 3. 11
- [84] MARINHO, L.B., HOTH, A., JÄSCHKE, R., NANOPOULOS, A., RENDLE, S., SCHMIDT-THIEME, L., STUMME, G. & SYMEONIDIS, P. (2012). *Recommender systems for social tagging systems*. Springer Science & Business Media. 3, 9
- [85] MARLOW, C., NAAMAN, M., BOYD, D. & DAVIS, M. (2006). Ht06, tagging paper, taxonomy, flickr, academic article, to read. In *Proceedings of the seventeenth conference on Hypertext and hypermedia*, 31–40, ACM. 3, 34
- [86] MARS LAND, S. (2015). *Machine learning: an algorithmic perspective*. CRC press. 9

## REFERENCES

---

- [87] MCCULLOCH, W.S. & PITTS, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, **5**, 115–133. 4
- [88] MCPARLANE, P.J., MOSHFEGHI, Y. & JOSE, J.M. (2014). Collections for automatic image annotation and photo tag recommendation. In *International Conference on Multimedia Modeling*, 133–145. 28
- [89] MIKOLOV, T., CHEN, K., CORRADO, G. & DEAN, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. 88, 89
- [90] MIKOLOV, T., SUTSKEVER, I., CHEN, K., CORRADO, G.S. & DEAN, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119. 88, 89
- [91] MIKOLOV, T., GRAVE, E., BOJANOWSKI, P., PUHRSCHE, C. & JOULIN, A. (2018). Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*. 91
- [92] MILLER, G.A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, **38**, 39–41. 26
- [93] MILLER, G.A. & CHARLES, W.G. (1991). Contextual correlates of semantic similarity. *Language and cognitive processes*, **6**, 1–28. 87
- [94] MURPHY, K.P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press. 9
- [95] NAIR, V. & HINTON, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814. 11
- [96] NGUYEN, H.T.H., WISTUBA, M., DRUMOND, R.L. & SCHMIDT-THIEME, L. (2017). Cnn-fm: Personalized content-aware image tag recommendation. *Archives of Data Science, Series A (Online First)*, **2**, 16 S. online. 6
- [97] NGUYEN, H.T.H., WISTUBA, M., GRABOCKA, J., DRUMOND, L.R. & SCHMIDT-THIEME, L. (2017). Personalized deep learning for tag recommendation. In *Advances in Knowledge Discovery and Data Mining*, 186–197, Springer International Publishing. 6
- [98] NGUYEN, H.T.H., WISTUBA, M. & SCHMIDT-THIEME, L. (2017). Personalized tag recommendation for images using deep transfer learning. In *Machine Learning and Knowledge Discovery in Databases*, 705–720, Springer International Publishing. 6
- [99] NOH, H., HONG, S. & HAN, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1520–1528. 68



- 
- [100] PAN, S.J. & YANG, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, **22**, 1345–1359. 17
  - [101] PANAHI, S., WATSON, J. & PARTRIDGE, H. (2016). Information encountering on social media and tacit knowledge sharing. *Journal of Information Science*, **42**, 539–550. 1
  - [102] PARKHI, O.M., VEDALDI, A., ZISSERMAN, A. *et al.* (2015). Deep face recognition. In *BMVC*, vol. 1, 6. 68
  - [103] PASCANU, R., MIKOLOV, T. & BENGIO, Y. (2013). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 1310–1318. 82
  - [104] PENNINGTON, J., SOCHER, R. & MANNING, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543. 89
  - [105] QIAN, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural networks*, **12**, 145–151. 98
  - [106] QIAN, X., LIU, X., ZHENG, C., DU, Y. & HOU, X. (2013). Tagging photos using users’ vocabularies. *Neurocomputing*, **111**, 144–153. 35, 82
  - [107] RAE, A., SIGURBJÖRNSSON, B. & VAN ZWOL, R. (2010). Improving tag recommendation using social networks. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, 92–99. 36
  - [108] RASCHKA, S. (2015). *Python machine learning*. Packt Publishing Ltd. 9
  - [109] RAWAT, Y.S. & KANKANHALLI, M.S. (2016). Contagnet: Exploiting user context for image tag recommendation. In *Proceedings of the 2016 ACM on Multimedia Conference*, 1102–1106, ACM. 50, 82
  - [110] REDMON, J. & FARHADI, A. (2016). Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*. 65, 69, 91
  - [111] REDMON, J. & FARHADI, A. (2018). Yolo3: An incremental improvement. *arXiv*. 69, 91
  - [112] REDMON, J., DIVVALA, S., GIRSHICK, R. & FARHADI, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788. 65, 69, 91
  - [113] REN, S., HE, K., GIRSHICK, R. & SUN, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99. 5, 13, 65, 69

## REFERENCES

---

- [114] REN, S., HE, K., GIRSHICK, R., ZHANG, X. & SUN, J. (2017). Object detection networks on convolutional feature maps. *IEEE transactions on pattern analysis and machine intelligence*, **39**, 1476–1481. 68
- [115] RENDLE, S. (2010). Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 995–1000. 17, 21, 36, 37, 45
- [116] RENDLE, S. (2012). Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, **3**, 57. 17, 37
- [117] RENDLE, S. & SCHMIDT-THIEME, L. (2010). Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, 81–90. 17, 21, 36, 44, 45
- [118] RENDLE, S., BALBY MARINHO, L., NANOPOULOS, A. & SCHMIDT-THIEME, L. (2009). Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 727–736. 17
- [119] RENDLE, S., FREUDENTHALER, C., GANTNER, Z. & SCHMIDT-THIEME, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 452–461. 17, 41
- [120] ROBERTSON, S. (2004). Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, **60**, 503–520. 87
- [121] RONG, X. (2014). word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*. 89
- [122] RUMELHART, D.E., HINTON, G.E. & WILLIAMS, R.J. (1985). Learning internal representations by error propagation. Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science. 11
- [123] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M. *et al.* (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, **115**, 211–252. 29
- [124] SALTON, G. & BUCKLEY, C. (1987). Term weighting approaches in automatic text retrieval. Tech. rep., Cornell University. 87
- [125] SCHÜTZE, H. (1992). Dimensions of meaning. In *Proceedings of the 1992 ACM/IEEE conference on Supercomputing*, 787–796, IEEE Computer Society Press. 87
- [126] SEN, S., LAM, S.K., RASHID, A.M., COSLEY, D., FRANKOWSKI, D., OSTERHOUSE, J., HARPER, F.M. & RIEDL, J. (2006). Tagging, communities, vocabulary, evolution. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, 181–190, ACM. 3

- 
- [127] SEVIL, S.G., KUCUKTUNC, O., DUYGULU, P. & CAN, F. (2010). Automatic tag expansion using visual similarity for photo sharing websites. *Multimedia Tools and Applications*, **49**, 81–99. 35
  - [128] SHAH, R.R., SAMANTA, A., GUPTA, D., YU, Y., TANG, S. & ZIMMERMANN, R. (2016). Prompt: Personalized user tag recommendation for social media photos leveraging personal and social contexts. In *Multimedia (ISM), 2016 IEEE International Symposium on*, 486–492, IEEE. 82
  - [129] SHASHUA, A. & HAZAN, T. (2005). Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, 792–799, ACM. 16
  - [130] SIGURBJÖRNSSON, B. & VAN ZWOL, R. (2008). Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, 327–336. 3, 34, 35
  - [131] SIMONYAN, K. & ZISSERMAN, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. 12, 13, 52, 64, 67, 69, 85
  - [132] SPARCK JONES, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, **28**, 11–21. 87
  - [133] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I. & SALAKHUTDINOV, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, **15**, 1929–1958. 14
  - [134] STATISTA (2018). Number of social media users worldwide from 2010 to 2021 (updated june 1, 2018). 2
  - [135] SYMEONIDIS, P. (2016). Matrix and tensor decomposition in recommender systems. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 429–430, ACM. 15
  - [136] SYMEONIDIS, P., NANOPOULOS, A. & MANOLOPOULOS, Y. (2008). Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems*, 43–50, ACM. 16
  - [137] SZEGEDY, C., LIU, W., JIA, Y., SERMANET, P., REED, S., ANGUELOV, D., ERHAN, D., VANHOUCKE, V., RABINOVICH, A. *et al.* (2015). Going deeper with convolutions. *Cvpr*. 13, 64, 65
  - [138] SZEGEDY, C., IOFFE, S., VANHOUCKE, V. & ALEMI, A.A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, vol. 4, 12. 12

## REFERENCES

---

- [139] TANG, K., PALURI, M., FEI-FEI, L., FERGUS, R. & BOURDEV, L. (2015). Improving image classification with location context. In *Proceedings of the IEEE international conference on computer vision*, 1008–1016. 65
- [140] TRANT, J. (2009). Studying social tagging and folksonomy: A review and framework. *Journal of Digital Information*, **10**. 22
- [141] TSOUMAKAS, G., KATAKIS, I. & VLAHAVAS, I. (2009). Mining multi-label data. In *Data mining and knowledge discovery handbook*, 667–685, Springer. 58
- [142] TUCKER, L.R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, **31**, 279–311. 16
- [143] TURIAN, J., RATINOV, L. & BENGIO, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, 384–394, Association for Computational Linguistics. 87
- [144] VAN HOUSE, N.A. (2007). Flickr and public image-sharing: distant closeness and photo exhibition. In *CHI'07 extended abstracts on Human factors in computing systems*, 2717–2722, ACM. 25
- [145] WAN, L., ZEILER, M., ZHANG, S., CUN, Y.L. & FERGUS, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 1058–1066. 5, 12, 14, 38
- [146] WANG, G., HOIEM, D. & FORSYTH, D. (2009). Building text features for object image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 1367–1374, IEEE. 65
- [147] WANG, J., YANG, Y., MAO, J., HUANG, Z., HUANG, C. & XU, W. (2016). Cnn-rnn: A unified framework for multi-label image classification. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, 2285–2294, IEEE. 50, 82, 107
- [148] WANG, M., NI, B., HUA, X.S. & CHUA, T.S. (2012). Assistive tagging: A survey of multimedia tagging with human-computer joint exploration. *ACM Computing Surveys (CSUR)*, **44**, 25. 3
- [149] WEI, Y., XIA, W., HUANG, J., NI, B., DONG, J., ZHAO, Y. & YAN, S. (2014). Cnn: Single-label to multi-label. *arXiv preprint arXiv:1406.5726*. 36, 50
- [150] WEISS, K., KHOSHGOFTAAR, T.M. & WANG, D. (2016). A survey of transfer learning. *Journal of Big Data*, **3**, 9. 17
- [151] WESTON, J., BENGIO, S. & USUNIER, N. (2011). Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, vol. 11, 2764–2770. 50

- 
- [152] WILLIAMS, H.T., MCMURRAY, J.R., KURZ, T. & LAMBERT, F.H. (2015). Network analysis reveals open forums and echo chambers in social media discussions of climate change. *Global Environmental Change*, **32**, 126–138. 1
- [153] WU, H.C., LUK, R.W.P., WONG, K.F. & KWOK, K.L. (2008). Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, **26**, 13. 87
- [154] WU, L., YANG, L., YU, N. & HUA, X.S. (2009). Learning to tag. In *Proceedings of the 18th international conference on World wide web*, 361–370, ACM. 35
- [155] XIANG, Z. & GRETZEL, U. (2010). Role of social media in online travel information search. *Tourism management*, **31**, 179–188. 1
- [156] XU, Z., FU, Y., MAO, J. & SU, D. (2006). Towards the semantic web: Collaborative tag suggestions. In *Collaborative web tagging workshop at WWW2006, Edinburgh, Scotland*. 3
- [157] YANG, J., YU, K., GONG, Y. & HUANG, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 1794–1801, IEEE. 65
- [158] ZEPHORIA (2018). The top 20 valuable facebook statistics (updated may 25, 2018). 2
- [159] ZHANG, M.L. & ZHOU, Z.H. (2006). Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, **18**, 1338–1351. 58

